取扱い説明書

基本操作
コマンドリファレンス

Reference Manual



VERSION 1.1.4

はじめに

このたびは、2D/3D 汎用 CAD システム「JDesignCAD」をご利用いただき、誠にありがとうございます。

30 年にわたる CAD 開発の歩みと開発背景

弊社代表は、パーソナルコンピューター向け CAD の黎明期である 1990 年代初頭より、30 年以上にわたり CAD 業界に携わってまいりました。建築専用 CAD の積算システム開発をはじめ、AutoCAD 向け作画システムの開発・サポート、電気機器メーカーにおける自社製 CAD システムの開発など、CAD の進化とともに歩みを続けております。
独立後も、多様な業種に向けた専用 CAD や周辺システムの受託開発を手がけ、その過程で培ったコア機能を汎用コンポーネントとして蓄積してまいりました。
昨今、機械・建築・製造・土木など幅広い分野において、かつて数多く存在した「国産 CAD」の選択肢が少なくなっています。2D CAD においては今なお親しまれているシステムもありますが、3D までカバーした国産システムは数えるほどしか残っていません。

JDesignCAD のコンセプト

こうした背景と、年々高額化する CAD 導入・維持コストの課題を受け、中小規模の設計会社様や個人設計者様が手軽かつ本格的に導入できるツールとして開発したのが「JDesignCAD」です。
自社製コンポーネントをベースに、2D/3D の設計作業をスムーズに行える環境を提供いたします。
本製品は、用途に合わせて「Free 版」と 「STD 版」 と 「Pro 版」の 3 つのエディションをご用意しています。

■ Free 版（無償）

2D 製図、2D ダイナミックブロックの作成・編集、完成した 3D ダイナミックブロックの操作、JDC 保存、STL 出力を利用できます。
主要 CAD 形式の読み込みも可能です。DWG・DXF などへの書き出しと解析は STD 以上で利用できます。

■ スタンダード版（有償）

FREE の機能に加え、2D・3D の応力解析・熱解析などと、他社 CAD 向けの書き出し機能（DXF / DWG / JWW / STEP / OBJ / SVG / SXF / IFC 等）を搭載しています。
3D 面動作の作成・編集、各種解析、IFC 部品出力を利用できます。対応範囲は各章を参照してください。

■ Pro 版（有償）

本書は Free・STD・Pro の操作を説明します。機能ごとの利用条件は各章とライセンス案内を確認してください。

■ 高いデータ互換性

Free 版・STD 版・Pro 版ともに、主要形式（DXF / DWG / JWW / STEP 形式）のインポートに対応しており JDesignCAD 独自形式の読み書きも各エディションで行えます。

■ 今後の展望

Pro 版の収益はコア性能の向上と新機能開発へ再投資し、ものづくりを支える CAD として継続的に改善してまいります。

本書の使い方と全体目次

本書は JDesignCAD 1.1.4 の統合コマンドリファレンスです。基本操作、形状編集、板金、ダイナミックブロック、解析、IFC・Revit、カスタマイズと MCP の設定・利用・開発を説明します。

Word 上では、目次や索引の項目から該当箇所へ移動できます。通常は Ctrl キーを押しながらクリックします。印刷して使う場合は、右端のページ番号を参照してください。

全体目次

はじめに 1

第 1 部 画面構成と基本設定 5

画面構成、選択・入力、座標系、スナップ、画層、モデルとレイアウトに加え、表示言語と単位設定を説明します。

アニメーションビュー　31

レイアウトにビューを増やす　35

3D モデルから三面図を作る　36

第 2 部 リボンコマンドリファレンス 51

索引からコマンド名を選び、各ページで機能、入力項目、操作手順、注意点、画面例を確認できます。収録順はリボンのタブとグループに沿い、末尾にレイアウトのコマンドをまとめています。

ファイル　ファイル操作 57

ホーム　作画 64

クロソイドを作図する　120

クロソイド基本型と横断線　121

ホーム　モデリング 131

ホーム　修正 164

ホーム　モデル修正 190

ホーム　注釈 224

挿入　挿入 248

挿入　ブロック定義 252

挿入　書き出し 255

挿入　参照 258

管理/計測　管理 259

管理/計測　計測 261

レイアウト　右クリックコマンド 268

第 3 部 ダイナミックブロック 276

第 4 部 解析機能 311

第 5 部 IFC 部品と Revit ファミリ 353

第 6 部 インストールと更新 371

リボンと表示モード　45

第 7 部 板金コマンドリファレンス　374

第 8 部 カスタマイズガイド　477

第 9 部 MCP 連携ガイド　495

第 10 部 API メソッドと操作のリファレンス　510

93 操作の入力と 152 データ型　554556

ショートカットを登録する　47

大判用紙の印刷プレビュー　48

画面フィットと複雑な図形の表示　49

ソリッドの体積を計算する　50

複数エッジのフィレットと接続を確認する　206

突起と穴を保ちながら面を移動する　213

曲線を面へ投影する　219

切断と干渉チェックの結果を確認する　203

画面例と設定値について

画面例は操作説明用です。操作が共通する画面を継続収録しています。配色、表示文字、ファイル履歴などは利用環境によって異なります。

目的の操作を探す

本書は現行版の操作を機能別にまとめています。初めて使う方は第 1 部から、目的の機能を調べる方は目次から参照してください。

表示言語・モデル単位・固定単位・部品の単位 40
3D ダイナミックブロック・数式・可視状態 277
解析の選び方・個別材料・梁と熱の実習 312
IFC 部品と Revit 2026／2027 ファミリ 353
操作別の参照先 372
配布物と編集用原本の管理 373

本書の表記

画面に表示される操作名は［ ］で示します。数値例は操作の理解と入力確認のための例で、実製品の設計条件や許容値を指定するものではありません。

エディションの確認

機能	Free	STD／Pro
2D ダイナミックブロックの作成・編集	利用可	利用可
完成した 3D ダイナミックブロックの操作	利用可	利用可
3D 面動作の作成・編集	利用不可	利用可
解析タブの各機能	利用不可	利用可
IFC ライブラリ書き出し	利用不可	利用可
SXF 読み込み	利用可	利用可
SXF 保存・DWG／DXF 等の書き出し	利用不可	利用可

日本語／英語の表示切替と、モデルの長さ単位は独立した設定です。解析・穴・IFC では、各入力欄に明記された単位へ合わせてください。

第 1 部

画面構成と基本設定

第 1 部は、JDesignCAD で作図を始める方に向けた基本操作編です。画面各部の役割、図形の選び方、設定の変更場所、モデルとレイアウトの使い分けを説明します。まず画面と入力操作を確認し、次に自分の作業に必要な設定を整えてください。

個々の作図・編集コマンドの詳しい入力方法は、第 2 部「リボンコマンドリファレンス」を参照してください。第 1 部では、複数のコマンドに共通する操作と設定を扱います。

対象は JDesignCAD 1.1.4 です。画面構成の図とリボン案内を現行版へ更新しました。言語・単位の設定は本部 25〜28 も参照してください。

目次

01 起動画面から図面を開く 6
02 操作画面の構成 7
03 マウスとキーボードの基本操作 8
04 図形を選択する 9
05 コマンドの実行と数値入力 10
06 ツールバーのファイル操作と表示操作 11
07 座標系と作業平面 13
08 スナップとトラッキング 15
09 レイヤと作画属性 17
10 右クリックメニューの使い分け 19
11 プロパティエクスプローラーの四つのタブ 20
12 プロジェクト設定の項目 21
13 レイヤとスタイルを整える 23
14 全体設定の保存と入力と選択 25
15 全体設定の表示と補助図形 26
16 全体設定のスナップとグリッドと近似 27
17 全体設定の読み書きと印刷 28
18 モデルとレイアウトの関係 29
19 モデルビューの設定 30
20 レイアウトの用紙とビュー尺度 32
21 レイアウトの表示とロックと稜線 33
22 レイアウトに図を配置する 34
23 寸法と中心線を入れて印刷する 38
24 初回設定と困ったときの確認 39

01 起動画面から図面を開く

起動画面では、新規図面の作成、既存図面の読み込み、最近使ったファイルの確認を行います。作図中にファイル関係の操作を行う場合も、上部の「ファイル」タブを使用します。

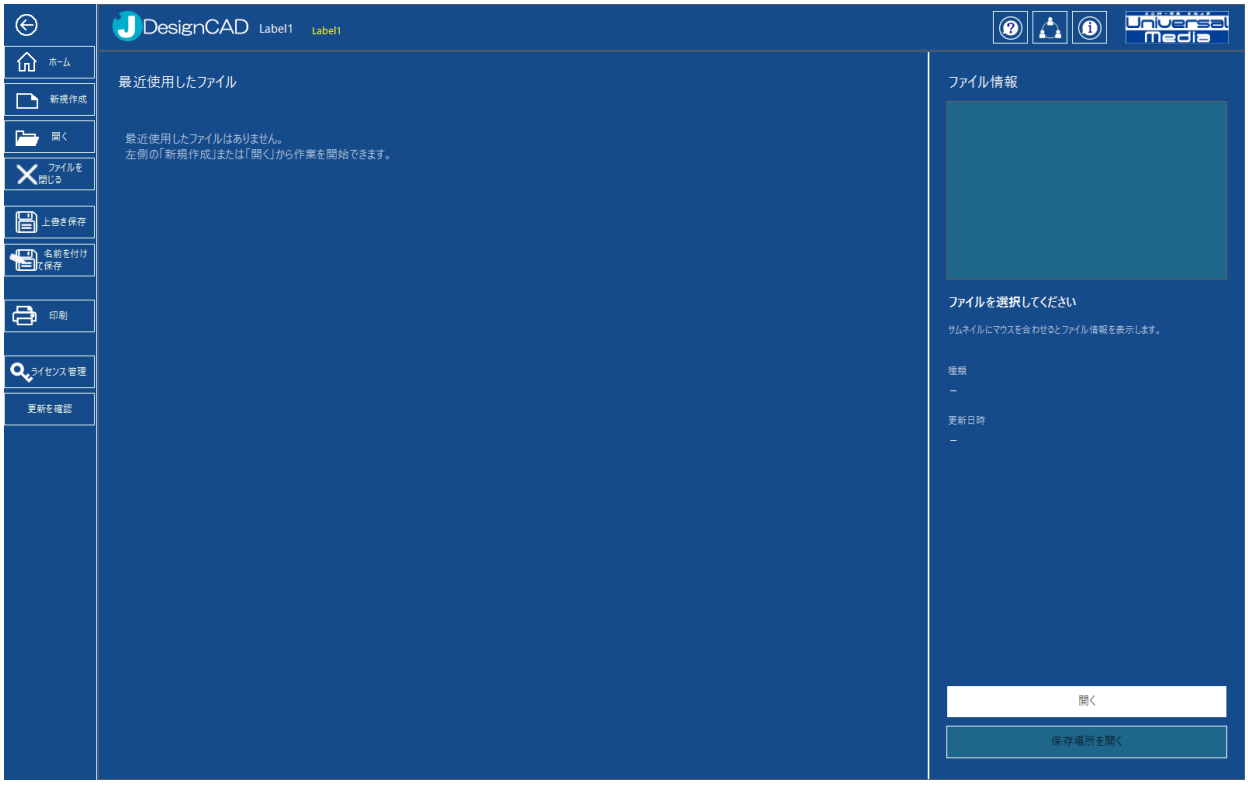


図 1 起動画面の例。左側がファイル操作、中央が最近使ったファイル、右側が選択ファイルの情報です。

新しい図面を作る

1. 左側の「新規作成」を選択します。
2. 操作画面が表示されたら、作図するモデルビューと作業平面を確認します。
3. 最初の保存では「名前を付けて保存」を選び、保存場所と図面名を指定します。継続編集には JDC 形式を使用します。

保存済みの図面を開く

「開く」からファイルを指定します。最近使ったファイルに表示されている場合は、対象を選び、右側のプレビューやファイル情報を確認して「開く」を押します。「保存場所を開く」では、そのファイルがあるフォルダーを確認できます。

保存と終了

「上書き保存」は現在の名前・場所へ保存します。修正前を残す場合は「名前を付けて保存」で別名にします。「ファイルを閉じる」で保存確認が表示されたら、変更を残すか確認してください。「印刷」では出力設定を開き、用紙や尺度をプレビューで確認します。

最近使ったファイルの一覧は図面データそのものではありません。ファイルを移動・削除した場合は、実際の保存場所から開き直してください。

02 操作画面の構成

操作画面は、コマンドを選ぶ部分、図形を操作する部分、数値や設定を編集する部分に分かれています。操作に迷ったら、まず右側の入力欄と最下部のメッセージを確認します。

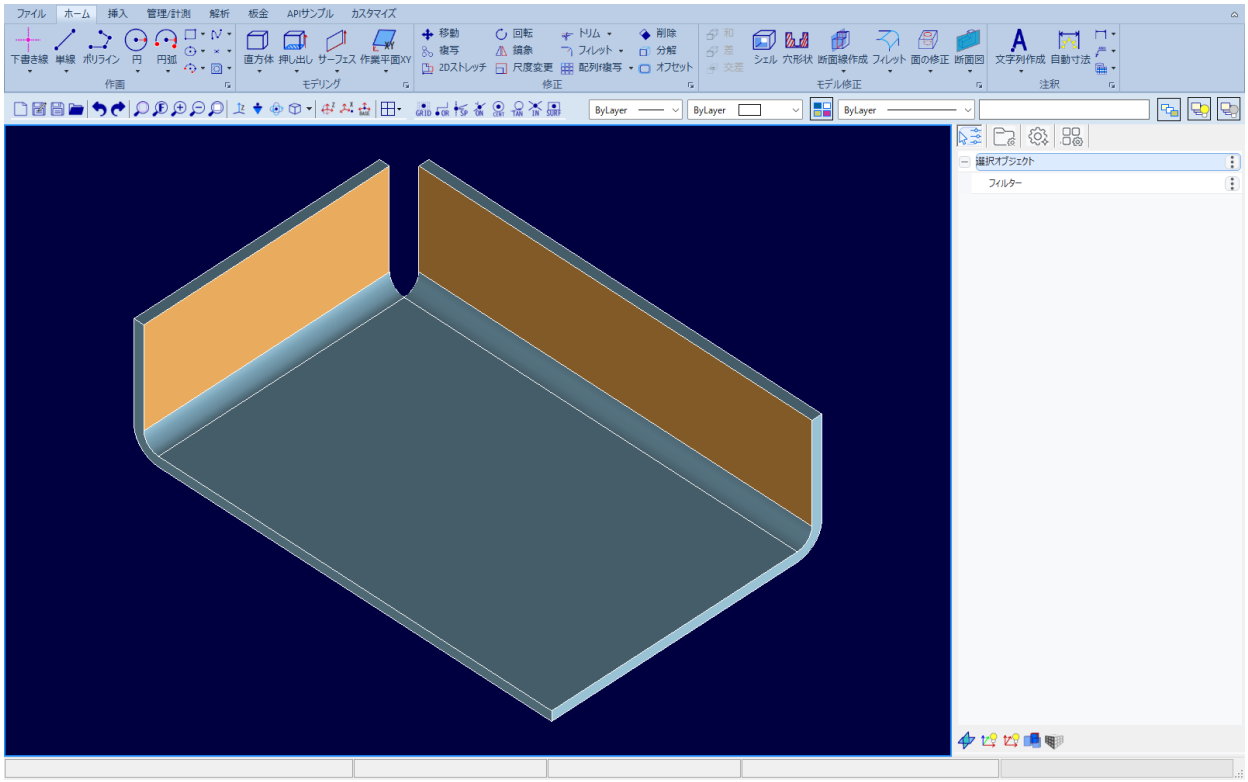


図 2 操作画面の全体。各部の位置は下表を参照してください。

画面の部分	位置と役割
リボンタブ	画面上部。「ホーム」「挿入」「管理／計測」などへ切り替え、目的のコマンドを選びます。「ファイル」では保存・印刷などを行います。
アイコンツールバー	リボンの下。ファイル操作、表示操作、スナップ、レイヤや色など、頻繁に使う項目をまとめています。
図形エディター	中央の広い領域。図形の作成・選択、位置指定、プレビュー確認を行います。モデルビューまたはレイアウトビューを表示します。
プロパティエクスプローラー	画面右側。実行中のコマンドの入力、選択図形の属性、プロジェクト設定、全体設定、ビュー設定を扱います。
ステータスバー	画面最下部。次に必要な操作のメッセージ、座標、処理の進行状況などを表示します。

コマンドが薄い色で表示されるとき

図形を選択していない、対象がソリッドではない、レイアウトでは使用できないなど、現在の条件では実行できない場合があります。コマンド名だけでなく、作業しているビューと選択対象を確認してください。

選択、入力待ち、作業平面などの色は設定や操作状態で変わります。色だけで判断せず、対象の名前や入力欄も確認します。

03 マウスとキーボードの基本操作

図形の編集と画面の移動を区別すると、意図しない変形を防げます。画面を移動したいときは、図形やハンドルを左ドラッグせず、ホイールボタンを使用します。

操作	主な働き	使用時の注意
左クリック	図形の選択、点・位置の指定、コマンドや入力欄の選択	実行中のコマンドによって、求められる対象が変わります。
左ドラッグ	範囲選択、選択図形やハンドルの操作	ドラッグ開始位置に図形があるか、空白かを確認します。
ホイール回転	ポインター付近を基準に画面を拡大・縮小	モデルの実寸やレイアウトの印刷尺度を変更する操作ではありません。
ホイールボタンを押してドラッグ	画面の平行移動	拡大中に別の場所を確認するときに使います。
Ctrl を押してホイールボタンをドラッグ	モデルビューの視点回転	作業平面を変更する操作とは別です。
右クリック	対象や操作状態に応じたメニューを表示	図形上、空白、ビュー枠、プロパティ項目で内容が異なります。

確定とキャンセル

作図中は、プロパティエクスプローラー上部の「✓」で確定し、「×」で中止します。Enter や Esc が使える操作では、ステータスバーにも案内が表示されます。連続作図やビュー追加では、終了しても既に作成した図形が残る場合があります。

数値欄を編集中の Enter は、その数値の入力確定として働きます。コマンド全体が終了したかどうかは、プレビューと操作メッセージを確認してください。

同じコマンドをもう一度使う

コマンドが終了または中止され、図形エディターが入力を受け付ける状態で Enter または Space を押すと、直前の繰り返し可能なコマンドを再開します。前回の図形選択をそのまま再処理するのではなく、新しい入力から始まります。

削除、切り取り、ファイル操作などの一回限りの操作は、この繰り返しの対象外です。ビュー専用コマンドは元の対象ビューを参照するため、別のビューを操作するときはそのビューからコマンドを選び直します。

間違えた操作を戻す

ツールバーの「アンドゥ」で直前の編集を戻し、「リドゥ」で戻した編集をやり直します。入力途中の取り消しは、まずコマンドの中止を使用してください。アンドゥは保存用のバックアップの代わりにはなりません。

04 図形を選択する

通常の選択では、対象をクリックして選びます。複数の図形をまとめて扱う場合は Ctrl を使うか、空白から矩形の範囲を指定します。編集中のコマンドがある場合は、そのコマンドが必要とする種類の図形だけが選択対象になります。

選択方法	操作	選ばれる対象
単独選択	図形を左クリック	クリックした図形。通常は前の選択を置き換えます。
追加と解除	Ctrl を押して図形をクリック	選択済み集合へ追加します。選択済み図形を再指定すると解除します。
ウィンドウ選択	空白から左→右へ矩形をドラッグ	矩形の内側に完全に入った図形を選びます。
交差選択	空白から右→左へ矩形をドラッグ	矩形の内側の図形と、矩形にかかる図形を選びます。
範囲を追加	Ctrl を押しながら範囲選択	既存の選択を残して追加します。
重なった図形の切り替え	Alt を押して同じ付近をクリック	重なりの候補を切り替えます。選択表示を確認して確定します。

選択前に確認すること

1. 不要なコマンドが実行中であれば中止し、通常の選択状態へ戻します。

2. 対象が表示されているレイヤと、レイヤのロック状態を確認します。

3. 必要に応じて右クリックの選択モードや、プロパティのフィルターを確認します。

4. 選択後は、ハイライトとプロパティに表示された図形の種類を確認します。

ソリッドと面と稜線

ソリッド全体、ソリッドの面、稜線は別の対象です。例えばソリッド移動とフィレットでは、選ぶ単位が異なります。見た目が同じ線でも、独立した 2D 線、3D モデルの稜線、レイアウトの投影輪郭では使えるコマンドが異なります。

選べないときの確認順

まず対象を拡大し、選択モード・フィルター・レイヤを確認します。レイアウトでは、ビュー枠を操作しているのか、ビュー内の稜線を指定するコマンドを使っているのかも確認してください。画面に見えている輪郭を、常に独立した線として編集できるわけではありません。

05 コマンドの実行と数値入力

多くのコマンドは、リボンで起動し、右側の入力欄を順番に指定して実行します。画面上のプレビューは入力内容の確認用です。プレビューが表示されただけでは、図形の作成や編集が完了していない場合があります。

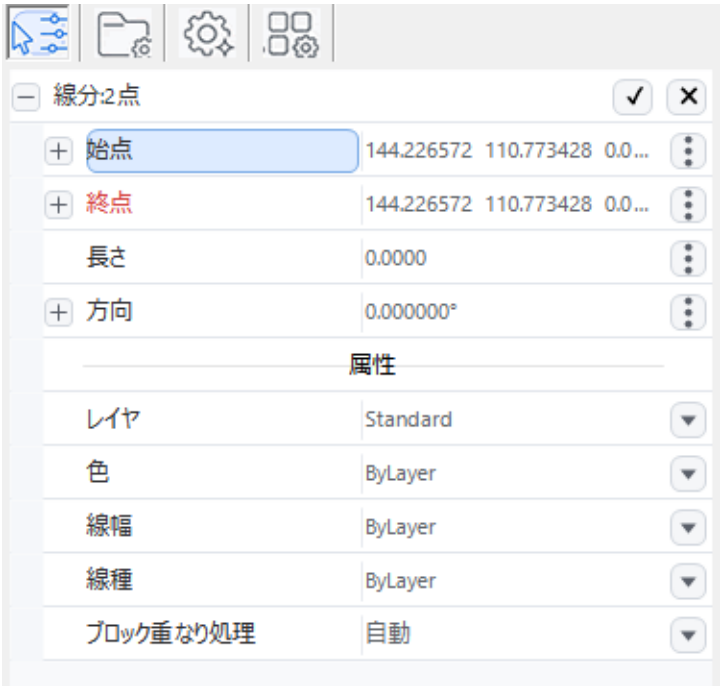


図 3 線分の始点と終点を指定する入力パネルの例。上部右端の✓と×が確定・中止です。

基本的な流れ

1. リボンでコマンドを選びます。小さな▼があるボタンは、展開すると関連する作図方法を選べます。

2. 右側の入力欄で、現在指定する点・要素・数値を確認します。

3. 点や要素は図形エディターで指定します。数値を正確に決める場合は値の欄を選び、入力して確定します。

4. プレビューで位置、向き、大きさ、演算モードなどを確認します。

5. 必須の入力をそろえ、「✓」またはそのコマンドの確定操作で実行します。

パネルの表示	読み方
左側の項目名と右側の値	左が何を指定するか、右が現在の値です。座標や方向は展開すると成分を確認できます。
＋と－	下位の入力項目を開く・閉じる操作です。項目を折りたたんでも入力値は残ります。
▼や縦の点のボタン	選択肢、追加メニュー、要素指定などを開きます。表示される内容は項目ごとに異なります。
赤い入力項目	入力を求めている項目の目印です。必要な指定はステータスバーでも確認します。

半径と直径など、同じ寸法を別の指定方法で入力する項目もあります。必要な方を選んで入力し、表示される値とプレビューが意図どおりか確認してください。

06 ツールバーのファイル操作と表示操作

リボンの下のアイコンツールバーには、よく使う操作が並んでいます。アイコンの上にポインターを置くと、名称を確認できます。ボタンの並びは画面幅や構成によって異なる場合があります。

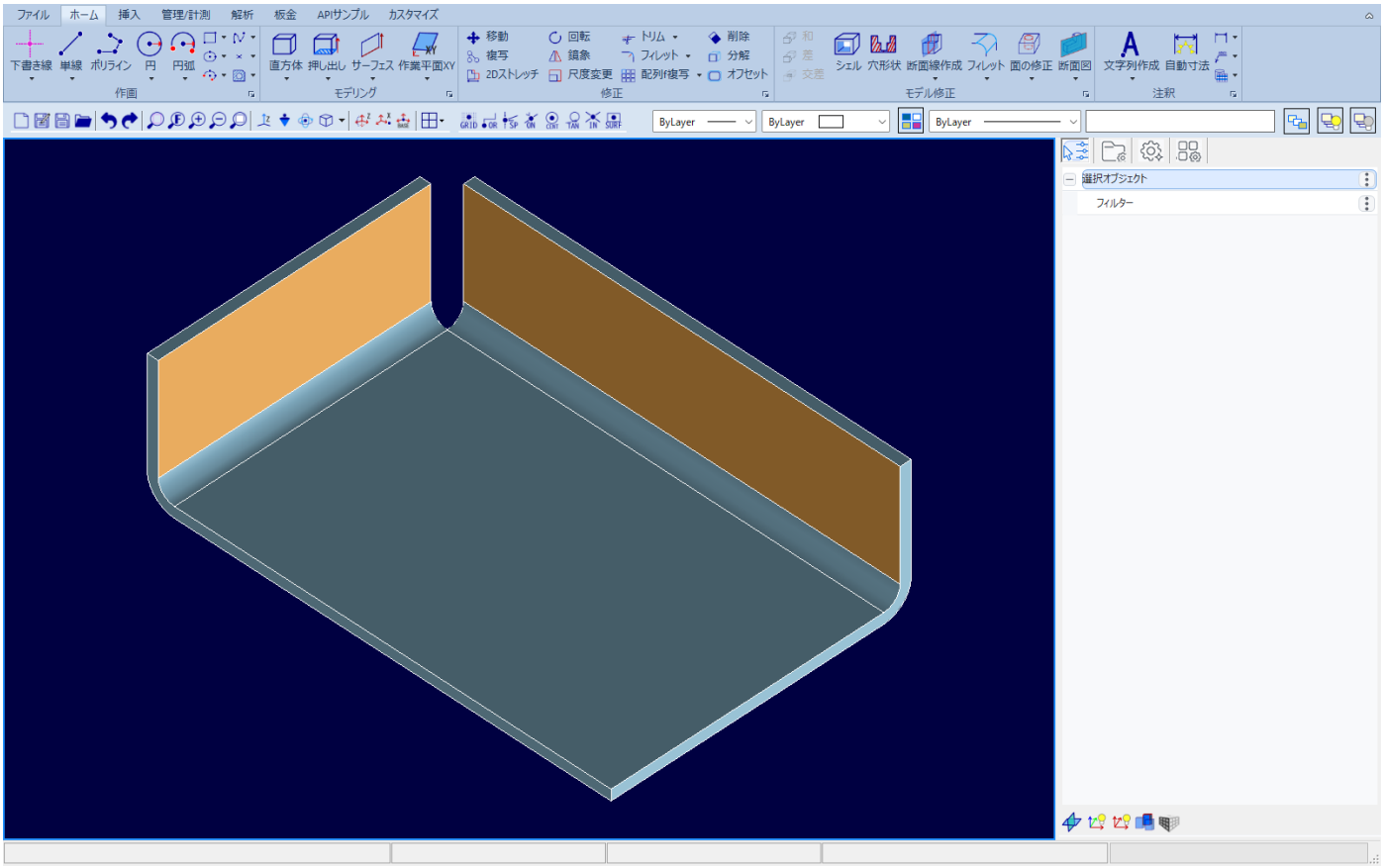


図 4 ツールバー左側。ファイル操作から表示操作、作業平面、スナップへ続きます。

コマンド	役割
新規作成／開く	新しい図面を作成する、保存済みの図面を開く操作です。
名前を付けて保存／保存	別名保存と、現在のファイルへの保存です。修正前の状態を残す場合は別名保存します。
アンドゥ／リドゥ	編集を戻す、戻した編集をやり直す操作です。
再表示	画面を描き直します。残像などの表示を確認するときに使用します。
画面フィット	図形の全体が見えるように表示範囲を調整します。
拡大／縮小	画面の表示倍率を段階的に変えます。
対角拡大	対角の 2 点で指定した範囲を拡大表示します。
Z 軸方向を上向き	視点の向きを整え、Z 軸が上向きとなる表示に調整します。
視点回転の中心点	回転表示の基準位置を指定します。小さな部分を中心に観察するときに便利です。
視点回転移動	マウス操作で視点を変更します。図形自体を回転する編集とは異なります。
視点方向	上面、正面、右面、左面、各方向のアイソメ、作業平面などへ切り替えます。

コマンド	役割
モデルビュー分割	同じモデルを 1〜4 画面に分けて確認します。レイアウトの三面図作成とは別の機能です。

拡大したら図形が見つからなくなったとき

まず「画面フィット」を使用します。それでも図形が小さすぎる場合は、遠くに離れた図形や、表示対象のレイヤが含まれていないか確認します。図形を見つけるために実寸を拡大する必要はありません。

表示だけを変更する操作

ズーム、平行移動、視点方向の変更ではモデルの形状や寸法は変わりません。ただし、レイアウト内でビューの尺度や投影方向そのものを変更した場合は、紙面上の図の見え方が変わります。操作対象を区別してください。

07 座標系と作業平面

WCS は図面全体の基準となる座標系、UCS は現在の作業に合わせた座標系です。作業平面を側面や傾斜面へ変更すると、UCS の X・Y 方向と、平面に垂直な Z 方向も変わります。

作図する平面を先に決める

2D の線や円を 3D モデルの側面に作る場合は、先に作業平面を設定します。視点を正面や側面にしただけでは、作業平面まで切り替わったとは限りません。作業平面と UCS マークを表示して向きと原点を確認すると確実です。

操作や表示	役割
作業平面の選択	リボンの作業平面コマンドから、標準面や要素を基準とする平面を設定します。詳細は別冊の作業平面コマンドを参照します。
UCS の Z 軸反転	作業平面に対する表裏の向きを反転します。押し出し方向などを確認するときに使用します。
UCS の X 方向	作業平面内の X 方向を指定します。
UCS XY 原点設定	作業平面上で扱う原点を設定します。
視点方向の作業平面	現在の作業平面を見やすい方向から表示します。
ステータスバーの WCS／UCS	同じ位置を、全体の基準と現在の作業基準で確認します。

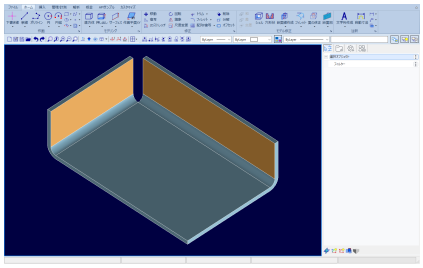


図 5 プロパティエクスプローラー下部の表示切り替えボタン。

補助表示の切り替え

下部のボタンでは、作業平面、WCS マーク、UCS マークの表示・非表示を切り替えられます。「アクティブレイヤー以外を透過する」は、作業対象を見分けるための表示です。表示を隠しても、座標系や図形そのものが削除されるわけではありません。

等角投影／透視投影の切り替えは見え方の変更です。寸法を読み取るための正投影図と、形状を立体的に見せる透視図を使い分けてください。

トラッキングの方向

現在の作業平面を基準に、平面内の方向と平面に垂直な方向を判断します。Z 方向の青いトラッキング線は UCS の Z 方向を表します。側面の作業平面では、WCS の Z 方向と一致しない場合があります。

WCS から UCS を作る

XY の WCS が有効な状態でも、座標系ツールバーの［Z フリップ］［UCS 原点］［X 方向］を使用できます。X 方向では現在の原点から方向を示す点を指定します。極トラッキングを有効にすると、設定した方向へ追跡できます。キャンセル時は元の座標系を保ちます。

08 スナップとトラッキング

スナップは、図形の端点や中心などを正確に指定するための補助機能です。画面上で近くをクリックするだけではなく、目的の点にスナップしたことを確認して入力します。

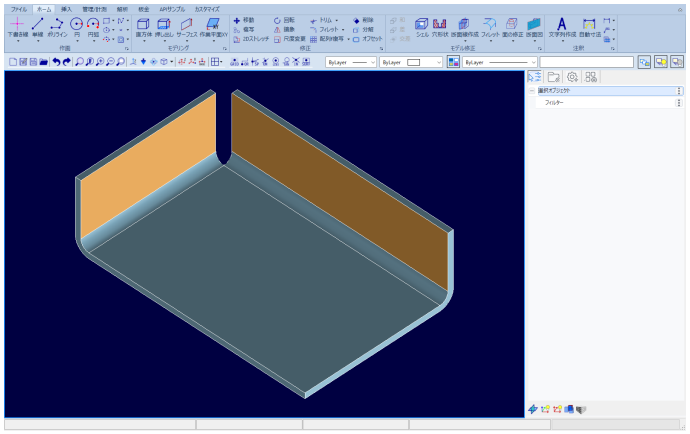


図 6 スナップの切り替えボタン。必要な種類を有効にして使用します。

ボタンの名称	使用する場面
グリッド	設定された格子点に合わせて点を指定します。グリッドの表示とは別の切り替えです。
直交	基準点から直交方向を使って作図するときに使用します。
端点 オブジェクトスナップ点	線の端点など、その図形が持つ特徴点を指定します。
要素上点	線や曲線の途中など、要素上の点を指定します。
中心点	円や円弧などの中心を指定します。円周上の点とは異なります。
接点	接する位置を利用する入力で使用します。候補は実行中のコマンドと図形により異なります。
交点	要素同士の交点を指定します。
フェース上点	3D モデルの面上の位置を指定します。

必要なスナップだけを有効にする

候補が多いと、近くにある別の点へ吸着することがあります。例えば中心間の寸法では中心点、端点からの作図では端点を優先し、不要な種類を一時的に切ると指定しやすくなります。

トラッキングを使う

「全体設定」→「スナップ設定」で、極トラッキング、極トラッキング角度、オブジェクトスナップトラッキング、Z 方向トラッキングを設定します。補助線を利用して、既知の点からの方向や位置を合わせます。

極トラッキング角度は方向の候補を決める設定であり、作成済みの図形を回転する設定ではありません。補助線が意図しない方向に出る場合は、作業平面と有効なスナップを確認します。

グリッドが見えても吸着しない場合

格子の「表示」と「グリッドスナップ」は独立しています。格子を見せる設定に加えて、ツールバーのグリッドスナップを有効にしてください。間隔はグリッド設定の水平距離・垂直距離で確認します。

09 レイヤと作画属性

色、線種、線幅を図形ごとに指定することも、レイヤの設定に従わせることもできます。図面全体の見た目をそろえる場合は、用途別のレイヤを用意し、図形の属性を ByLayer にすると管理しやすくなります。

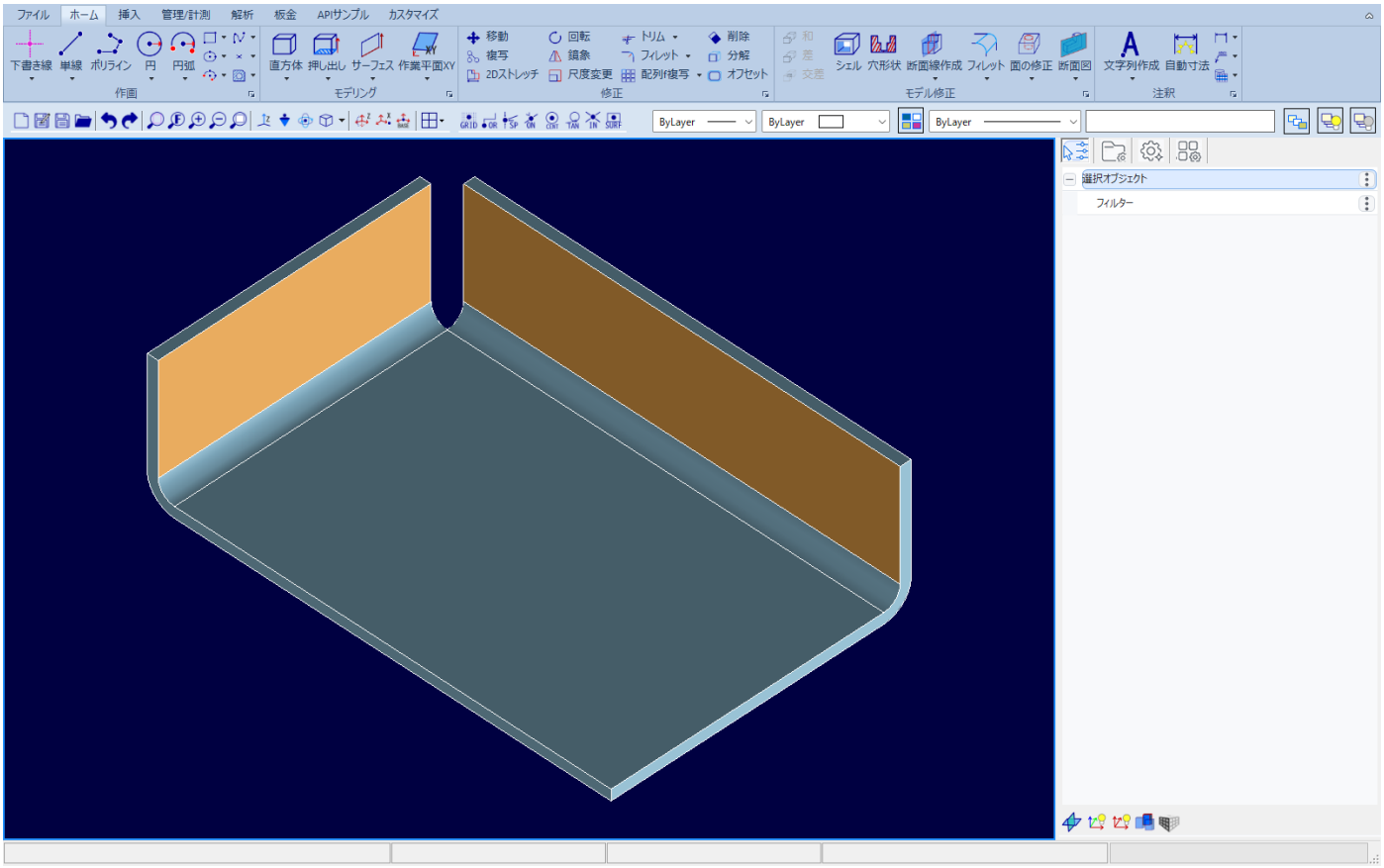


図 7 ツールバー右側の属性欄。レイヤ、色、線幅、線種などを確認します。

属性	役割
レイヤ	図形を用途別に分類します。外形、寸法、中心線、補助図形などを分けて管理できます。
色	図形の表示色を指定します。印刷時に色をそのまま出すかは、印刷設定でも確認します。
線種	実線、破線、一点鎖線などのパターンを指定します。
線幅	図形の線の太さを指定します。表示モードや印刷設定によって見え方が異なります。
ByLayer	色・線種・線幅を、所属するレイヤの設定から引き継ぎます。

新しく描く図形と選択済み図形

何も選択していない状態の属性欄は、これから作成する図形の設定に使います。既存図形を選択している場合は、その図形の属性変更になることがあります。新規作画の設定だけを変えたいときは、先に選択を解除してください。

用途別レイヤで作図する手順

1. 「プロジェクト設定」→「レイヤテーブル」で必要なレイヤを用意します。

2. 各レイヤの色、線種、線幅を設定します。

3. 作図するレイヤを現在のレイヤにし、色・線種・線幅を ByLayer にします。

4. テスト用の線を 1 本描き、所属レイヤと属性を確認します。

5. 既存図形もそろえる場合は、対象を選択してレイヤと ByLayer 属性を設定します。

レイヤの色を変更しても、個別に固定色を指定した図形の色は変わりません。線種や線幅も同様です。「レイヤ設定が反映されない」ときは、図形側の属性を確認します。

プロジェクト設定でレイヤを追加・改名すると、ツールバーのレイヤ一覧にも反映されます。レイヤ管理では一覧とイメージを切り替え、表示・ロック・現在レイヤ・色・線種・線幅を設定できます。表示状態はアクティブビューへ適用します。

中心線コマンドの属性

中心線コマンドで新規作成する線は、「中心線」レイヤに色・線種・線幅を ByLayer として配置されます。レイヤがなければ寸法線と同じ色、線種 CENTER で作成します。既存の「中心線」レイヤがある場合、その設定は変更しません。

10 右クリックメニューの使い分け

右クリックメニューは、クリックした場所と選択状態に応じて変わります。図形の編集、ビューの操作、プロパティ項目の追加設定など、対象に直接関係するコマンドを探すときに使用します。

右クリックする場所	主な操作	確認する点
図形エディターの空白	選択モード、貼り付け、表示関係など	コマンド実行中か通常選択中かで内容が変わります。
選択した図形	切り取り、コピー、削除、移動、回転、尺度変更、分解など	選択図形の種類によって使える項目が変わります。
ソリッドの面や稜線	面・稜線を対象とする編集や作成	ソリッド全体が対象か、一部の面や稜線かを確認します。
レイアウトのビュー	投影ビューの追加、稜線の表示、中心線、ビュー表示やロックなど	必要なビューを対象にしてから実行します。
プロパティの項目	項目の追加・削除、選択肢、設定固有の操作	親の一覧を操作するのか、1 件の項目を操作するのかを確認します。

基本的な操作

1. 通常選択または目的のコマンドの状態で、対象を選びます。

2. 対象の上で右クリックし、表示されたメニューから操作を選びます。

3. コマンドが始まったら、プロパティの入力とステータスバーの案内に従います。

4. 不要なメニューは閉じます。項目が無効の場合は、対象や選択条件を確認します。

レイアウトでよく使う操作

ビューの「稜線を非表示」は、そのビューの図面表現から不要な稜線を除く操作です。3D モデルの稜線を削除する操作とは異なります。「選択稜線の太さを変更」も、モデルの属性を変更せず紙面上の太さを調整します。

ビュー自体を隠す場合は「ビューを表示」を切り替えます。隠したビューを戻す操作と、そのビュー内で非表示にした稜線を戻す操作は別です。

メニューが想定と違うとき

選択を解除して対象を選び直すか、現在のコマンドを中止してから右クリックします。レイアウトでは、紙面上に直接描いた線と、投影ビュー内のモデルの輪郭を区別してください。

11 プロパティエクスプローラーの四つのタブ

右側のプロパティエクスプローラーは、上部の四つのアイコンで表示内容を切り替えます。特に「現在の図面だけを変える」のか、「アプリケーションの共通設定を変える」のかを区別してください。

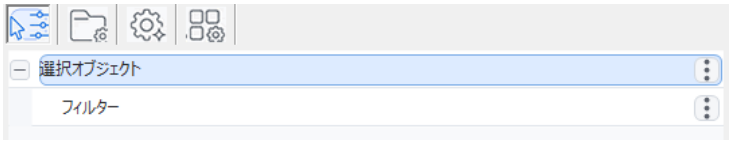


図 8 左から選択と操作、プロジェクト設定、全体設定、ビューの順に並ぶタブ。

タブ	扱う内容	主な反映先
左端の選択と操作	選択した図形の情報、実行中のコマンドの入力	選択図形、または今から実行する操作
2 番目のプロジェクト設定	色、レイヤ、線種、寸法スタイルなど	現在開いている図面
3 番目の全体設定	操作方法、表示色、スナップ、自動保存、出力設定など	アプリケーションの共通動作。属性一覧は新規図面の初期設定にも使用
右端のビュー	モデルビューとレイアウトの一覧、視点、用紙、ビュー尺度など	指定したビューまたはレイアウト

左端のタブの読み方

何も選択していない場合は選択オブジェクトの一覧やフィルター、図形を選んだ場合はその種類や属性が表示されます。コマンドを起動すると、同じ領域がそのコマンドの入力パネルに切り替わります。

例えば線を選択した状態では端点や属性を確認し、円の作図中では中心・半径などを入力します。パネルの最上段の名称を見て、「既存図形を編集している」のか「新しい図形を入力している」のかを確認します。

設定項目を操作する

左端の＋で展開し、項目名の右の値を選んで編集します。▼のある欄は候補を選び、縦の点や右クリックでは追加操作を確認します。数値を入力した後は Enter などで入力を確定します。

見たい項目がない場合は、正しいタブを開いているか確認し、親項目を展開してスクロールします。「全体設定」の一覧は長いため、下の方にも設定が続きます。

入力パネルを自動で表示する

「全体設定」→「操作」→「操作パネル自動表示」を確認します。作図を始めたときに入力項目が見つからない場合は、左端のタブへ切り替えてください。

12 プロジェクト設定の項目

プロジェクト設定は、現在の図面に登録されている属性やスタイルを管理するタブです。図面の保存時にこれらの情報も保存されます。他の図面を開いた場合は、その図面が持つ設定を確認してください。

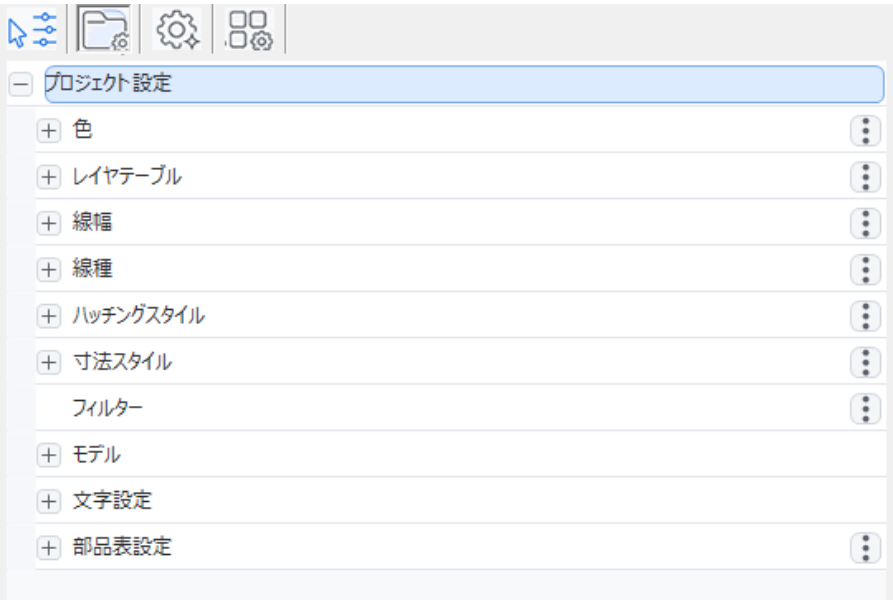


図 9 プロジェクト設定の一覧。各項目を展開すると登録内容を確認できます。

項目	設定する内容
色	図面で使用する色の一覧。既存の色の確認や必要な色の登録に使います。
レイヤテーブル	レイヤの一覧と各レイヤの属性。用途別分類や表示・編集の管理に使います。
線幅	図面で使う線幅の一覧。必要な太さを登録します。
線種	実線、CENTER、HIDDEN などの線パターン。図形やレイヤへ適用する定義です。
ハッチングスタイル	塗りつぶしやハッチングのパターンと属性を管理します。
寸法スタイル	寸法の文字、矢印、線、表示方法などをまとめて管理します。
フィルター	図形の種類や属性による選択条件を管理します。
モデル	図面内のモデルを管理します。視点を管理する「ビュー」とは異なります。
文字設定	文字サイズ、フォント、太字・斜体などの文字書式を設定します。
部品表設定	部品表で使用する列などの構成を設定します。

現在の図面と新規図面を分けて考える

この図面の線種や寸法スタイルを変更したい場合は、プロジェクト設定を使います。新しく作る図面の初期の属性一覧を整える場合は、全体設定側の同名の一覧を確認します。全体設定の一覧を変えただけで、既存図面がすべて置き換わるわけではありません。

部品の色・線種を挿入先へ合わせる

ブロック内部の設定	挿入後の属性
画層 0＋ByLayer	挿入先画層の色・線種・線幅
画層 0＋ByBlock	配置したブロック参照の属性
画層 0 以外＋ByLayer	元の画層の属性
明示的な色・線種・線幅	指定した値を維持

汎用部品を挿入先の色に合わせるなら、内部を画層 0・ByLayer で作成します。画層 0 は予約画層なので、名前変更・削除・PURGE による除去はできません。ブロック外の画層 0 の図形は、画層 0 自身の属性を使います。

13 レイヤとスタイルを整える

作図前に、レイヤと代表的なスタイルを少数に整理しておくと、後から線の見た目や寸法をそろえやすくなります。既存図面を編集する場合は、その図面のルールを先に確認してください。

レイヤを用意する

1. プロジェクト設定の「レイヤテーブル」を展開します。

2. 一覧のメニューから新しいレイヤを追加し、用途が分かる名前を付けます。

3. レイヤの色、線種、線幅を設定します。作図できない場合はロックや表示の状態も確認します。

4. ツールバーで作図先のレイヤを選び、図形の属性を ByLayer にして作図します。

レイヤ名の例	設定方針の例
外形	実線を使い、主となる輪郭の太さを統一します。
寸法	外形と区別できる色や線幅にそろえます。寸法スタイルの指定も確認します。
中心線	線種 CENTER を使います。既存の中心線レイヤがある場合はその設定を尊重します。
補助図形	仕上げの前に表示を切り替えやすいように分けます。

これらは分類例です。社内の製図ルールに合わせて名前と属性を決めてください。レイヤを非表示にすることと、図形を削除することは異なります。

線種と線幅を確認する

「線種」で使いたいパターンが登録されているか確認し、レイヤまたは図形に割り当てます。見た目だけで判断せず、プロパティの線種名も確認します。線幅を使う場合は、モデルビューの線幅モードと印刷設定も合わせて確認します。

線幅を追加するには、プロジェクト設定の「線幅」で新規項目を作り、名前と「線幅(mm)」の数値を入力します。例えば名前を「0.5mm」、線幅(mm)を 0.5 にします。名前だけを「0.5mm」にしても、実際の幅は 0.5 mm になりません。

追加・名前変更・数値変更は属性プルダウンへ反映されます。幅が 0 の独自項目は名前と 0mm を表示します。ByLayer はレイヤ、ByBlock は挿入ブロックの設定を使用します。線種の名前やパターンを変更した場合も一覧へ反映されます。

寸法と文字の見た目をそろえる

「寸法スタイル」を展開し、使用するスタイルの文字、矢印、線などを確認します。用紙上で読める大きさになっているか、代表的な寸法を 1 つ作成して確かめてから全体へ適用します。

通常の文字は「文字設定」で文字サイズと書式を確認します。文字と寸法は別の設定を持つため、一般の文字サイズを変更するだけで寸法文字も同じになるとは限りません。

既存図面を変更するとき

スタイルやレイヤの変更は、それを参照する複数の図形へ影響することがあります。まず別名で保存し、変更後はモデルだけでなくレイアウトと印刷プレビューも確認します。

14 全体設定の保存と入力と選択

全体設定は、JDesignCAD を使うときの共通動作を調整するタブです。上部には新規図面の初期値に使う色・レイヤ・線幅・線種・ハッチング・寸法・文字・部品表の設定があり、その下に操作や表示などの設定が続きます。

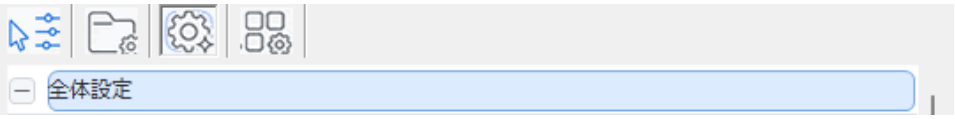


図 10 全体設定タブ。下に属性一覧、自動保存、表示、スナップ、印刷などの設定が続きます。

自動保存

「自動保存を有効にする」で有効・無効を切り替え、「保存間隔（分）」と「保存先」を指定します。保存先は、空き容量があり、書き込み可能で、後から探しやすいフォルダーを選んでください。

自動保存は、変更のある図面のバックアップを時刻付きの JDC ファイルとして保存します。元のファイルへの上書き保存とは別です。作図中や処理中は保存を待つ場合があり、図形のない図面や変更のない図面では保存が行われない場合があります。区切りごとの手動保存も続けてください。

表示形式

「小数区切り」と、一般値・座標・ベクトル・角度の小数桁数を設定します。「半径／直径」は円に関する入力表示の使い分けに使用します。小数桁数を減らしても、モデル形状をその桁数へ丸める設定ではありません。寸法の表示形式は寸法スタイル側も確認します。

選択と操作

項目のまとまり	調整できる内容
選択の大きさと感度	ハンドルサイズ、ドラッグ開始距離、選択枠表示を調整します。小さい図形を編集しにくい場合に確認します。
選択の色と見え方	選択枠、ハンドル、選択・フォーカスのハイライト色、ハイライト線幅・不透明度・表示位置を調整します。
事前ハイライト	選択前の候補表示とその色を設定します。選択済みの表示と混同しない色にします。
操作パネル自動表示	コマンド開始時の入力パネル表示を設定します。
ソリッド作成後に 2D スケッチを削除	ソリッド作成後に元の 2D スケッチを残すかどうかに関係します。元断面を残して再利用したい場合は確認してください。

15 全体設定の表示と補助図形

表示の設定は作業の見やすさを整えるものです。背景や補助表示の色を変えるときは、図形、選択色、トラッキング線を見分けられる組み合わせにしてください。

座標マークと色設定

「表示」ではワールド座標マークと UCS 座標マークの表示や位置を確認します。「色設定」では、背景色、グリッド色、作図補助色、レイアウト背景色、作業平面色、作業平面の透明度、アクティブビュー枠色を変更します。

項目	調整する場面
背景色／レイアウト背景色	モデル空間と紙面の見やすさを調整します。図形そのものの色はレイヤ・図形属性で変更します。
グリッド色／作図補助色	格子や入力補助を、実際に作成する図形から見分けやすくします。
作業平面色／透明度	作業平面の位置を確認しつつ、背後のモデルも見えるように調整します。
アクティブビュー枠色	分割画面や対象ビューの枠を背景から見分けやすくします。

下書き線

「下書き線」では長さ係数、線幅、色を設定します。位置合わせや作図の基準として使う補助線の見た目を整える設定です。下書き線を作成・消去するコマンドや表示切り替えは、第 2 部の下書き線の説明を参照してください。

下書き線と中心線は用途が異なります。図面に残す中心線は中心線コマンドで作成し、専用レイヤと線種を使用します。

その他の補助設定

設定グループ	内容と使いどころ
干渉チェック	干渉ソリッド色、接触面色、それぞれの結果のレイヤを設定します。検査結果を元のモデルから見分けるために使います。
ブロック属性	「挿入時に属性入力ダイアログを表示」を設定します。属性付きブロックの挿入手順に関係します。
バルーン マーカー	初期記号、接頭文字、尾文字、連番方式、既定属性を設定します。部品番号などの注記に使います。
フォント表示	表示モードと精度を調整します。図面で使用する文字のフォントや文字サイズとは別の設定です。

16 全体設定のスナップとグリッドと近似

点の指定に関する設定と、曲線を近似する設定は目的が異なります。選択しにくい、方向が合わないという問題を、近似精度やパスの最大ギャップで調整しないようにしてください。

スナップ設定

項目	働き
30 度スナップ／45 度スナップ	対応する角度の候補を使う入力補助を設定します。
作業平面原点にスナップ	現在の作業平面の原点を点指定の候補にします。
ワールド原点にスナップ	WCS の原点を点指定の候補にします。UCS 原点と一致するとは限りません。
極トラッキング	基準から方向を合わせる補助を有効にします。
極トラッキング角度	方向候補に使う角度を設定します。
オブジェクトスナップトラッキング	図形のスナップ点を利用して位置合わせの補助を行います。
Z 方向トラッキング	作業平面に垂直な方向の位置合わせを補助します。

端点・中心点・交点などのスナップ種類の有効・無効は、上部ツールバーでも切り替えます。まず必要な種類を選び、追加の方向補助が必要な場合にトラッキングを使用します。

グリッド

「水平距離」と「垂直距離」で格子の間隔を設定し、「表示」で格子の見せ方を選びます。モデルビューにもグリッド設定があるため、現在の画面を調整する場合は「ビュー」タブの対象モデルビューの値も確認します。

グリッド間隔は作図時の補助です。例えば 10 の間隔を選んでも、図面の寸法がすべて 10 の倍数に変更されるわけではありません。グリッドへ吸着させるにはグリッドスナップを有効にします。

近似とパス

設定	意味と注意
近似の精度	曲線などを近似するときの精度を設定します。数値表示の小数桁数や、すべてのソリッド演算の精度を一括で変える設定ではありません。
曲線の近似方法	近似処理で使用する表現方法を選びます。元の曲線を変換する操作では、結果を確認してから採用します。
パスの最大ギャップ	要素をつなげてパスにする際に扱う隙間の基準です。離れた別要素を誤ってつながないよう、必要以上に大きくしません。

通常の作図を始める段階では、近似や最大ギャップを不用意に変更せず、必要になった処理に限って試してください。変更前の図面は別名で残します。

17 全体設定の読み書きと印刷

外部形式とのやり取りでは、相手側のソフトや図面の用途に合わせて設定します。編集用の JDC ファイルを残し、交換用ファイルは別に出力すると、元の情報を確認しやすくなります。

書き出し設定と STEP 読み込み

項目	確認する内容
DXF DWG 形式と書き出しバージョン	出力形式と相手側が読み込めるバージョンを選びます。
DXF DWG 代替フォントと文字	フォントの置き換えや文字の取り扱いを確認します。出力後は日本語や記号の表示を確かめます。
寸法の読み込み／書き出し	寸法の取り扱いを設定します。外部ソフトでの表示と編集結果も確認してください。
STEP のマルチスレッド	読み込み処理の並列実行に関する設定です。
STEP の類似面を結合	読み込み時の面の整理に関する設定です。面や稜線の構成が変わる場合があります。
STEP のブロックを構成	読み込んだデータをブロックとして構成する設定です。
STEP の非周期サーフェスを優先	読み込み時の面の表現方法に関する設定です。問題のあるデータを比較するときは、元ファイルを保持します。

印刷

項目	役割
印刷方式	使用する描画・出力方式を選びます。方式により使用される下位設定が異なります。
色の出力	カラーなどの出力方法を確認します。画面の背景色とは別です。
規定線幅の出力幅	規定値の線幅を出力するときの幅を設定します。
OpenGL 線種ピッチ倍率	OpenGL 方式での線種の繰り返し間隔を調整します。
GDI 線種ピッチ倍率	GDI 方式での線種の繰り返し間隔を調整します。
GDI シェーディング	GDI 方式での陰影付き表示の扱いを設定します。
GDI ビットマップ印刷	GDI 方式での画像化した印刷の扱いを設定します。
GDI 重なり処理／表示順	図形が重なる部分の出力や描画順を調整します。

18 モデルとレイアウトの関係

モデルは実寸で作図・編集する図形データです。モデルビューは、そのモデルを見るための作業画面です。レイアウトは、モデルを必要な方向と尺度で用紙上に配置し、図面として仕上げる場所です。

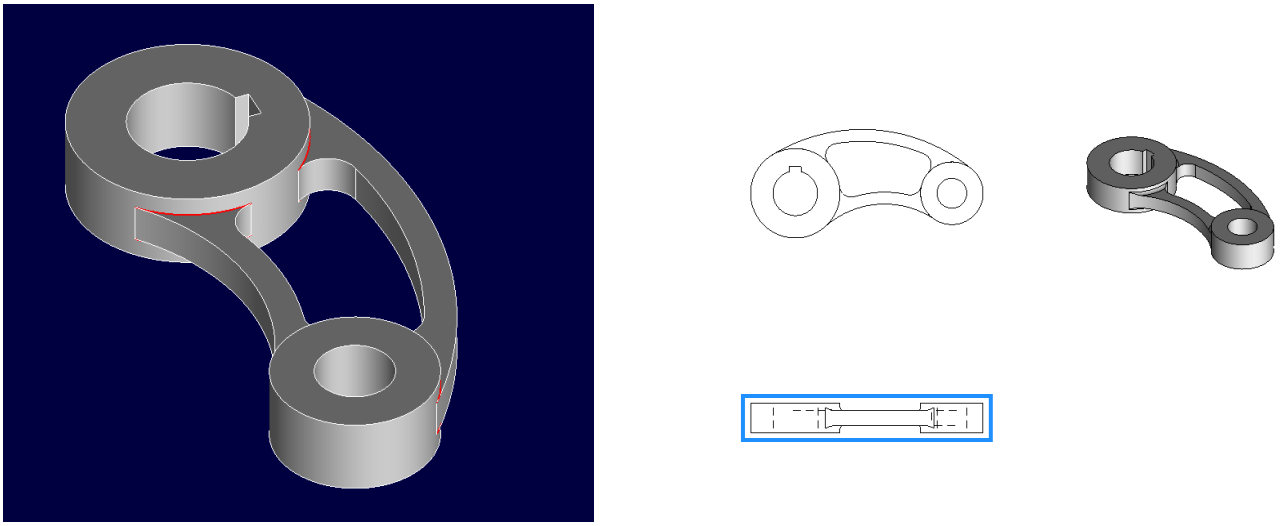


図 11 モデルビューの例（左）とレイアウトの例（右）。左の赤は稜線の選択表示です。

比較する点	モデルとモデルビュー	レイアウト
主な目的	図形を実寸で作成し、形状や位置を編集する	用紙、投影方向、尺度、注記を整え、出力する
画面の拡大縮小	見る範囲が変わる。実寸は変わらない	用紙の見え方が変わる。ビュー尺度は別に設定する
同じ形状の表示	同じモデルを複数の方向から観察できる	同じモデルから正面図、上面図、詳細図などを配置できる

投影ビューは元モデルを参照して表示します。モデルの変更は関連する図へ反映されますが、寸法や中心線の追従には参照の関連付けが必要です。用紙に直接描いた線や文字は、モデルとは別の紙面要素として扱います。

一つのモデルから複数の図を作る

同じモデルを正面から見た図、上から見た図、立体的に見た図として配置できます。紙面上の図を増やすために、部品のモデルそのものを複写する必要はありません。それぞれのビューに表示方法や尺度を設定できます。

用紙に直接描く図形

表題、注記、補足の線などは、レイアウトの紙面上に配置します。これらはモデルの形状とは別の要素です。投影図を移動するときは、紙面の注記が追従するか、関連付けの有無を確認してください。

19 モデルビューの設定

「ビュー」タブで対象のモデルビューを展開すると、その画面の視点や作業平面、表示方法を設定できます。図面内のモデルの管理はプロジェクト設定、見る方向の管理はビュー設定と覚えると区別しやすくなります。



図 12 モデルビューを展開した状態。

項目	役割
投影方向／距離	モデルを見る方向や視点の位置関係を設定します。数値の直接変更より、まず標準の視点方向コマンドで向きを確認します。
作業平面	現在の作図基準の平面を確認・設定します。視点方向だけとは別の設定です。
線幅モード	線幅を使用するかなど、画面上の線幅の扱いを設定します。
光源の明るさ	陰影付き表示の明るさを調整します。材質やモデルの形状を変更する設定ではありません。
光源の位置	陰影の方向を選びます。凹凸や穴の形を確認しやすい位置にします。
グリッド	このモデルビューで使う格子の設定を確認します。
表示レイヤ	このビューで表示するレイヤを選びます。
表示モード	線による表示やシェーディングなど、モデルの見せ方を選びます。

分割ビューを使う

上部のモデルビュー分割から、左右・上下などの分割や標準 4 面を選べます。同じモデルを共有するため、別画面にモデルを複製する必要はありません。操作したい区画をクリックし、アクティブ枠を確認して作業します。

各区画の拡大縮小や視点は個別です。分割は作業中の画面構成であり、紙面の投影図を作る機能ではありません。レイアウトへ切り替えると単一表示へ戻ります。図面に残す投影図は、レイアウト側で作成してください。

アニメーションビューで動きを確認する

アニメーションビューは、モデルの部品を指定した軸や曲線に沿って動かして表示する機能です。力や質量から動きを計算する動力学解析ではありません。解析画面の「変形アニメーション」とも別の機能です。

モデルに固定側と可動側を用意し、ビュー一覧から［新規アニメーションビュー］を作ります。表示レイヤを確認し、ドライブ一覧で［新規曲線ドライブ］または［新規軸ドライブ］を作成します。曲線ドライブには経路となる曲線、軸ドライブには軸上の点と方向を設定します。

［移動する要素］へ可動部品を指定します。固定側は含めません。経路や軸と、動かす部品は別の指定です。例では直線経路に沿ってスライダだけを動かします。曲線の開始位置を 0 にし、移動方向を確認します。

プログラム一覧で新規プログラムを作成し、使用するドライブ、時刻、位置を登録します。例は 0 秒で位置 0、2 秒で位置 40、4 秒で位置 0 です。対象のプログラムをアクティブにしてから、ビューのメニューで再生を開始します。曲線の位置は経路に沿う距離です。

再生方法は 1 回、無限リピート、指定回数から選べます。停止すると元の配置へ戻ります。モデル・レイアウトなど別のビューへ切り替えると再生は停止します。JDC で保存し、再度開いたときは使うプログラムと可動部品を確認して再生します。部品どうしの干渉や接触を自動判定して止める機能ではありません。

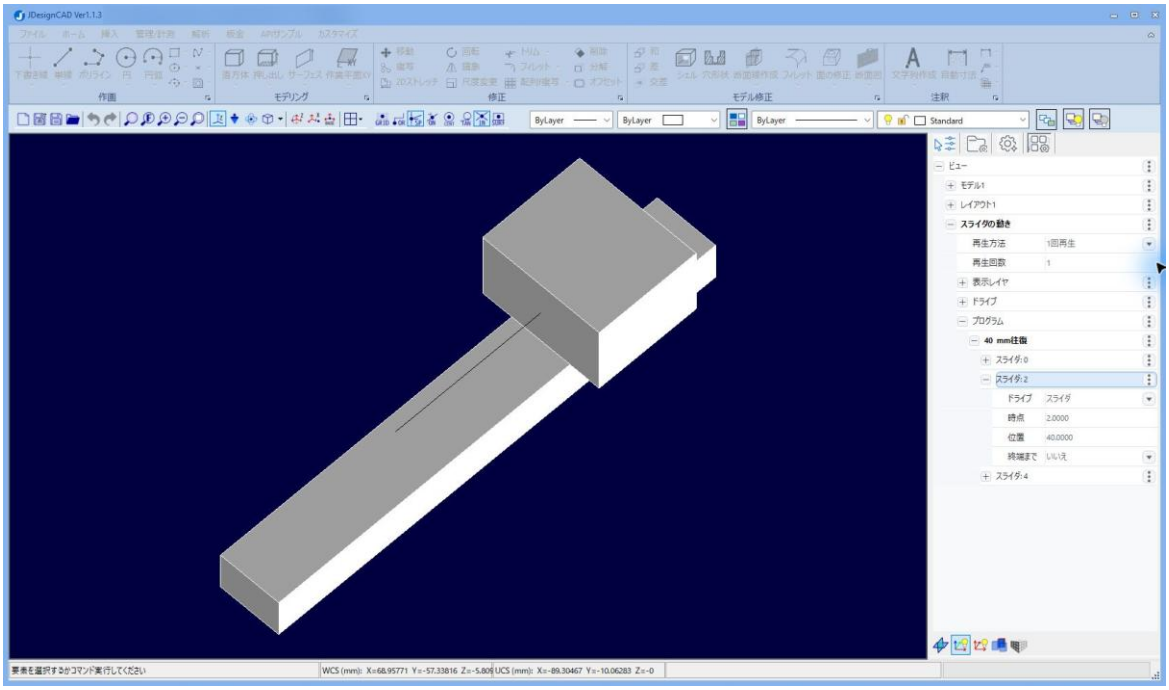


図 203　直線ドライブとスライダの設定例

20 レイアウトの用紙とビュー尺度

「ビュー」タブでレイアウトを展開すると、用紙とその中のビューを確認できます。用紙全体の設定と、個々のビューの設定は階層が異なります。

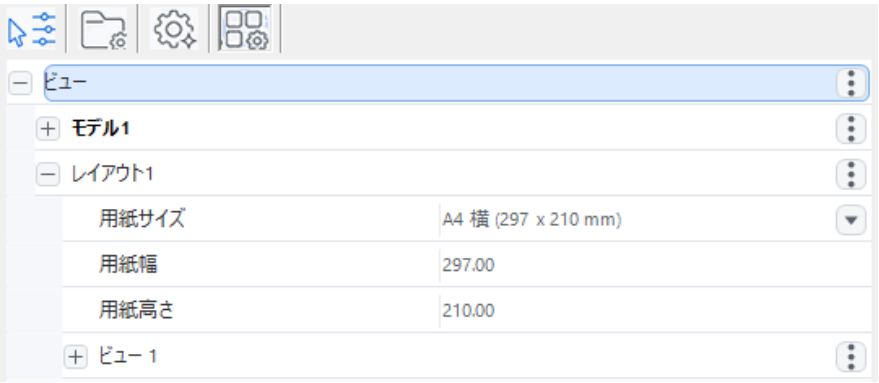


図 13 レイアウトの用紙設定。下の「ビュー 1」を展開すると個別ビューの設定が表示されます。

用紙を設定する

1. 対象のレイアウトを表示し、「ビュー」タブでそのレイアウトを展開します。

2. 「用紙サイズ」で用紙を選び、「用紙幅」「用紙高さ」を確認します。

3. 用紙に置く図の数、尺度、寸法や表題に必要な余白を考えて配置します。

4. 印刷時も同じ用紙と向きが選ばれているか確認します。

三つの大きさを区別する

大きさの種類	変更する場所	変更されるもの
モデルの実寸	作図やモデルの編集コマンド	部品や図形そのものの寸法
ビューの尺度	レイアウトの各ビューの「拡大縮小」	モデルを用紙上に表示する倍率
画面の表示倍率	ホイールや画面の拡大縮小	作業画面で見える大きさだけ

例えば実寸 100 mm の長さは、ビュー尺度 1:2 なら紙面上で 50 mm、2:1 なら 200 mm になります。関連付きの寸法はモデルの実寸を参照します。印刷側でも用紙を縮小すると、さらに別の倍率がかかるため注意してください。

ビューの位置と範囲

各ビューには、モデル側の表示位置、用紙側の位置、幅・高さの項目があります。「Left」「下」の後ろに付く「モデル」と「layout」を区別します。位置の微調整では、まずビュー枠と表示内容のどちらを移動したいかを決めてください。

枠の範囲外は表示されないため、図が欠けて見える場合はビュー範囲も確認します。尺度を変更しただけでモデル自体の寸法が変わることはありません。

21 レイアウトの表示とロックと稜線

レイアウトの個別ビューでは、表示方法や稜線の太さをモデルから独立して設定できます。形状を変えずに図面表現を整える場合は、このビュー側の設定を使用します。

項目	役割と注意
ビューを表示	ビューの表示・非表示を切り替えます。非表示ビューは印刷や 2D 書き出しの対象からも外れます。
ビューをロック	用紙上の位置、表示範囲、尺度、投影方向の変更を保護します。寸法用のスナップや紙面上の文字編集は可能です。
表示レイヤ	このビューに含めるレイヤを設定します。
投影方向と投影形式	見る方向と平行投影・透視投影などの形式を設定します。
明るさと光源の位置	陰影付きの図の見やすさを調整します。
表示モード	線による表示、陰影付き表示などを選びます。
図面の隠れ線を表示	図面表示で隠れ線を出すかを設定します。
隠れ線の線種と色	隠れ線のパターンと色を設定します。
枠線を表示	ビューの範囲を確認する枠の表示を切り替えます。
内側移動をロック	枠と表示内容の移動関係を変える設定です。「ビューをロック」による配置保護とは異なります。

稜線の太さを変える

ビュー全体は「稜線の太さ（mm）」へ紙面上の太さを入力します。0 は元のモデル属性を使う指定です。モデルのエッジ属性や他のビューには影響しません。

一部だけ変える場合は、「選択稜線の太さ（mm）」に値を入れ、項目またはビューの右クリックから「選択稜線の太さを変更」を実行します。稜線をクリックまたは矩形で選び、Enter で確定します。個別に 0 を適用すると、その稜線はビュー全体の設定に戻ります。

不要な稜線を隠す

ビューの右クリックから「稜線を非表示」を実行し、対象を選択して Enter で確定します。Esc はその選択操作の中止です。非表示は対象ビューにだけ適用され、モデルの稜線自体は削除されません。

ビューが見えないとき

非表示にしてもビューの設定はツリーに残ります。ビュー一覧から表示を戻すか、レイアウトの「すべてのビューを再表示」を使用します。手動で用紙に描いた文字や線はビューとは別なので、ビューを非表示にしても残る場合があります。

22 レイアウトに図を配置する

レイアウトには、基準となる図と、そこから派生する投影図、断面図、詳細図などを配置できます。完成形を印刷する用紙と尺度を先に決めると、後から配置を調整しやすくなります。

ベースビューから投影図を作る

1. 対象モデルを用意し、使用するレイアウトへ切り替えます。
2. レイアウトの右クリックメニューから「VIEWBASE（ベースビュー）」を実行します。
3. 向き、尺度、表示方法、第三角法または第一角法を確認します。
4. 用紙上をクリックしてベースビューを配置します。
5. 基準図から上下・左右などへ移動し、必要な投影図をクリックして配置します。
6. 必要な図を置いたら Enter または Esc で追加を終了します。基準図を配置する前の Esc は作成を中止しますが、配置後の終了では既に置いた図が残ります。

既存の図から追加する

後から図を増やす場合は、対象のビューを選び、右クリックの「投影ビューを追加」を使います。基準となるビューのモデル、尺度、投影法などを引き継いで配置します。元のビューと異なる図を参照していないか確認してください。

図の種類	使う目的
ベースビュー	正面など、図面の基準となる向きを配置します。
投影ビュー	基準図に対する上面・側面などを配置し、複数方向から形状を示します。
補助ビュー	傾斜した面など、通常の正面・上面では見えにくい方向を示します。
断面ビュー	切断した状態を表示し、内部形状を説明します。
詳細ビュー	部分を取り出して大きく表示し、小さな形状を説明します。

各ビューの作成時は、コマンドの入力欄とステータスバーの案内を確認してください。断面や詳細の見え方を変える操作と、元の 3D モデルを切断・拡大する編集は別です。

配置後の確認

ビュー名を分かりやすくし、用紙からはみ出していないか、図同士と寸法の間に余白があるか確認します。配置・尺度が決まったビューは「ビューをロック」で保護すると、注記中の誤操作を防ぎやすくなります。

レイアウトにビューを増やす

レイアウトを開き、プロジェクトのビュー一覧またはレイアウト上のメニューからビュー追加を実行します。モデル、向き、尺度、紙面上の範囲、表示レイヤを設定します。配置後はビュー名を付け、不要なレイヤを隠してから表示をロックすると、寸法や注記を入れる間の誤移動を防げます。

通常のビュー追加は、方向や尺度を独立して設定する窓を増やします。正面図から上下左右の投影図を配置したいときは、次のベースビューと投影ビューを使います。用紙上の空き位置へ図を置く操作と、モデル空間に別の 3D 部品を作る操作は区別してください。

3D モデルから三面図を作る

例は穴のある 80×50×20 mm の部品です。モデルの形状と穴を確認してレイアウトへ切り替え、用紙を A4 横に設定します。レイアウトの右クリックメニューから［ベースビュー］（VIEWBASE）を実行します。

モデル、正面方向、尺度 1:1、隠線の表示、第三角法を指定し、紙面の左下寄りに正面図を配置します。続いて上へ動かして平面図、右へ動かして右側面図を配置します。第三角法では平面図が上、右側面図が右です。第一角法では対応する配置側が逆になるため、図面で要求された投影法を先に決めます。

投影図を置き終えたら Enter または Esc で追加配置を終えます。配置済みのベースビュー・投影ビューから後で増やすには［投影ビュー］（VIEWPROJ）を使います。元ビューの尺度・投影法を引き継ぎます。通常の独立ビューや透視ビューを親として選ぶ操作ではありません。

寸法を入れる前に 3 図の向き、穴の隠線、位置合わせと尺度を確認します。寸法は投影された実形状を指定し、文字高さは印刷時の大きさで調整します。穴数・R・面取りなどの注記は文字付き引き出し線で付けます。モデルを変更した後はビューと寸法を見直し、印刷プレビューで線幅・文字・用紙内への収まりを確認します。

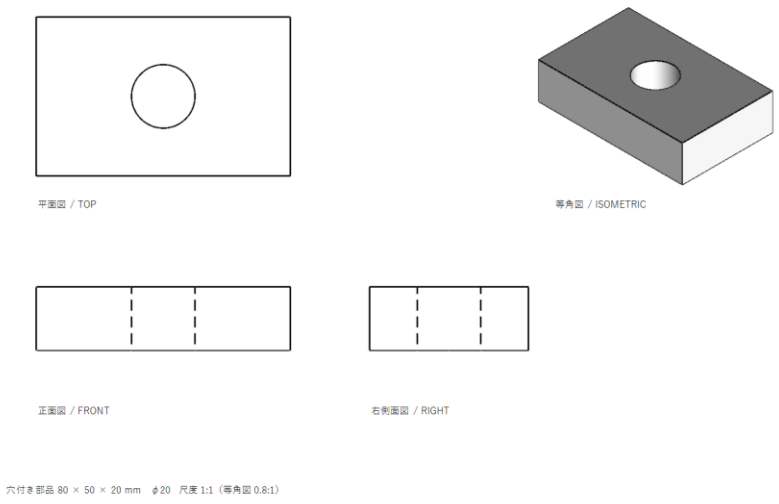


図 201　穴付き部品の三面図と等角図

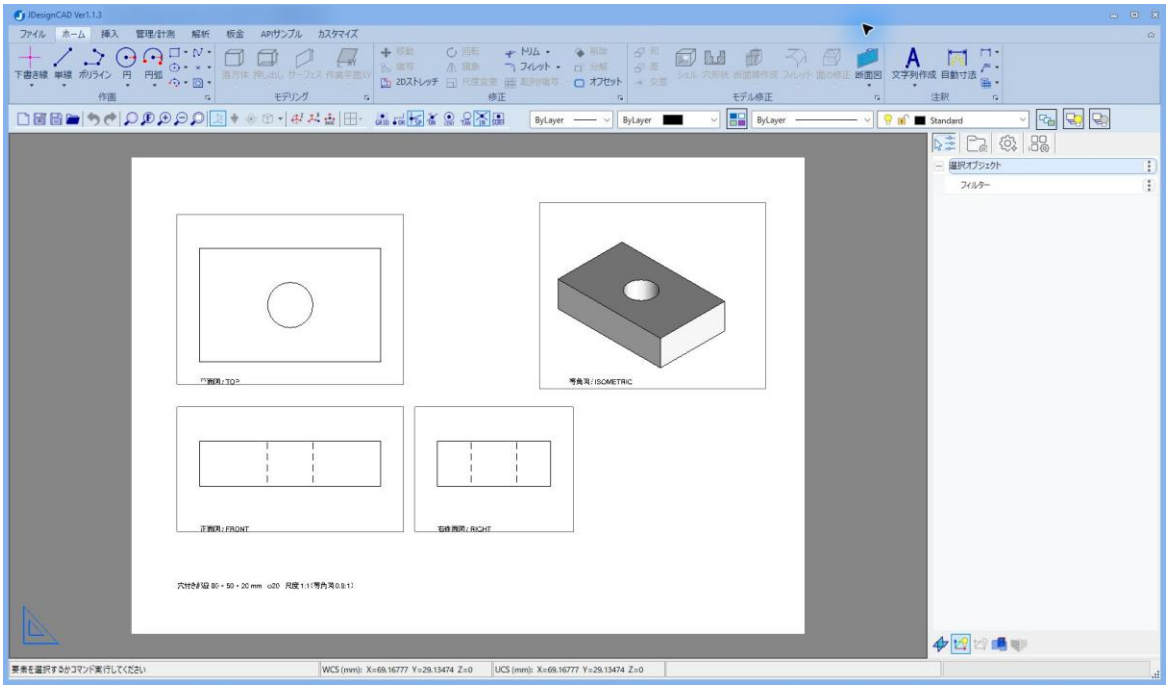


図 202　複数のビューを配置したレイアウト画面

23 寸法と中心線を入れて印刷する

レイアウトの寸法は、投影図の円や端点などを正しく参照させることが重要です。用紙上の見た目の位置だけを指定すると、モデルの実寸や変更への追従が意図どおりにならない場合があります。

寸法を入れる

1. ビューの向きと尺度を決め、必要ならビューをロックします。
2. 寸法コマンドを起動し、ビュー内の対象図形やスナップ点を指定します。
3. 円の中心間では中心点、直径・半径では該当する円や円弧を指定します。
4. 寸法を配置し、分かっているモデル寸法と一致しているか確認します。文字を上書きして数値だけ合わせる前に、参照先と尺度を確認してください。

中心線を入れる

「ホーム」→「単線」の▼から「円の中心線」または「2 線間の中心線」を選びます。円では対象の円・円弧を選び、2 線間では平行な 2 本の線を指定します。レイアウトでは同じビュー内の対象を選びます。

作成する対象	関連付けの扱い
レイアウトの投影図	元の円や 2 線とビューを参照する中心線を作成します。中心線にスナップして寸法を入れることもできます。
モデルビューの 3D 稜線	3D 稜線を参照する中心線を、その 3 次元位置に作成します。
通常の独立した 2D 円や線	関連付けのない中心線を作成します。元図形を変更しても自動追従しません。

レイアウトの「はみ出し長さ（紙面 mm）」は、ビュー尺度によらない紙面上の長さです。中心線は「中心線」レイヤへ ByLayer で作成されます。元の稜線の色や線幅を変更する必要はありません。

関連付きでも、参照元の削除や大きな形状編集で参照できなくなる場合があります。モデルを変更した後は、中心線と寸法が正しい位置・値になっているか確認します。

印刷前に確認する

1. 用紙サイズと向き、ビューの尺度、図と寸法のはみ出しを確認します。
2. 全体設定の印刷方式、色の出力、線幅・線種に関する設定を確認します。
3. 「ファイル」→「印刷」でプレビューを確認します。非表示ビューや隠した稜線が意図どおりかも確認します。
4. 実寸出力が必要な場合は、印刷側で用紙に合わせる縮小などが追加されていないか確認します。必要に応じて代表的な寸法を試し刷りで測ります。
5. JDC を保存し、交換用の図面が必要な場合は別に書き出します。

24 初回設定と困ったときの確認

使い始めは、作図に直結する項目から順に確認します。細かな近似・読み込み・印刷方式の設定は、必要な作業ができてから調整してください。

最初に確認する順番

1. 保存先と図面名を決め、まず JDC で保存します。

2. 全体設定の自動保存を確認し、保存先と間隔を決めます。

3. モデルビューと作業平面、UCS 原点・方向を確認します。

4. 必要なレイヤと線種を用意し、色・線種・線幅の ByLayer 設定を確認します。

5. 端点・中心点など必要なスナップを有効にし、グリッド表示と吸着を区別して設定します。

6. 背景、選択色、文字や寸法の見やすさを確認します。

7. 印刷する場合は用紙とビュー尺度を決め、代表的な線・文字・寸法をプレビューで確認します。

症状から確認する場所

症状	最初に確認すること
図形が選べない	実行中コマンド、選択モード、フィルター、レイヤの表示とロック、ビュー内の選択対象を確認します。
コマンドを押しても進まない	右側の必須入力、数値の確定、ステータスバーの指示を確認します。プレビューだけでは完了とは限りません。
設定を変えたのに図形へ反映されない	プロジェクト設定か全体設定か、図形の属性が ByLayer か固定値か、対象ビューの設定かを確認します。
グリッドが見えるが吸着しない	上部ツールバーのグリッドスナップを確認します。表示とは別です。
側面で点の方向や高さが合わない	視点だけでなく、作業平面と UCS の向き・原点を確認します。
レイアウトの図が動かない	ビューをロックしていないか、枠と内側のどちらを操作しているかを確認します。
寸法値が実寸と合わない	参照対象、円の中心点へのスナップ、ビュー尺度、寸法の関連付けを確認します。
印刷で線が太い、破線が詰まる	図形・レイヤ・ビューの線幅と、印刷方式・規定線幅・線種ピッチを順に確認します。
図面が見当たらない	画面フィット、表示レイヤ、非表示ビュー、実際のファイル保存場所を確認します。

変更した設定は、一つずつ結果を確かめてから次へ進めてください。個々のコマンドの入力順序や詳細なオプションは、第 2 部のリボンコマンドリファレンスを参照します。

言語・単位設定の目次

表示言語と図面単位を設定します。

25 表示言語を切り替える 41
26 モデルの入力・表示単位 42
27 単位が固定される入力欄 43
28 部品・外部形式の単位 44

25 表示言語を切り替える

リボン、起動時パネル、主なダイアログ、ツールチップを日本語／英語で表示できます。本書では日本語の操作名を使います。

設定手順

1. 作業中の図面を保存します。

2. プロパティエクスプローラーの［全体設定］で言語を日本語または英語へ変更します。

3. JDesignCAD を終了し、起動し直します。表示言語は再起動後に反映されます。

4. リボンと起動パネルの表示を確認し、図面を開いて作業を続けます。

言語と単位は別の設定

英語表示を選んでも、モデルが自動でインチになることはありません。モデルの単位は［プロジェクト設定］→［モデル］で設定します。反対に、インチで作図する場合も日本語表示のまま使用できます。

項目	言語を変えたときの扱い
リボン・主な設定画面	選択した言語で表示
図面名・部品名・画層名	利用者が付けた名前を保持
図面中の文字・追加の製品仕様	入力した文章を保持。自動翻訳しない
座標・長さ・角度・形状	表示言語だけでは変更しない
単位を明記した解析・IFC の欄	表示された単位に従う

画面例の読み方

本書の基本操作・コマンド図解には、操作が共通する旧版の画面例も使用しています。色、ボタンの位置、表示文字、ファイル履歴は現在の環境と異なる場合があります。操作と対応範囲は本文を基準にしてください。

26 モデルの入力・表示単位

モデルの単位は「長さをどの単位で入力・表示するか」を指定します。既存の形状を拡大・縮小する操作ではありません。通常のモデル座標は内部で mm として保持します。

設定手順と確認例

1. ［プロジェクト設定］→［モデル］→対象モデル（例：Standard モデル）を開きます。

2. ［単位］でミリメートル、センチメートル、メートル、キロメートル、インチ、フィートを選びます。

3. 長さや座標の表示を確認します。実寸 25.4 mm の線は、インチ表示では 1、メートル表示では 0.0254 になります。

4. 通常の長さ入力欄へ数値を入力すると、選んだモデル単位から実寸へ換算されます。インチ表示の通常の長さ欄へ 2 を入力すると 50.8 mm です。

種類	換算の考え方	25.4 mm 角の例
長さ・座標	単位の倍率で換算	一辺 1 inch
面積	長さ倍率の 2 乗	645.16 mm^2 = 1 $inch^2$
体積	長さ倍率の 3 乗	16,387.064 mm^3 = 1 $inch^3$
角度・個数・倍率・百分率	長さ換算しない	90 度、3 個、2 倍はそのまま

図面の大きさを本当に変えたい場合

実寸 100 mm の形状を 200 mm にする場合は、尺度変更や寸法に対応した編集を使います。単位を mm から cm へ切り替えても、100 mm が 10 cm と表示されるだけです。古い JDC 図面も従来の実寸を保って開きます。

紙面の尺度との違い

モデル単位、モデルの実寸、レイアウトの尺度は別々です。モデルをインチ表示にしても、用紙サイズや 1:10 などの印刷尺度は変わりません。モデル内・ブロック内の寸法表示はモデル単位を引き継ぎます。

27 単位が固定される入力欄

入力前に、欄の名称と単位を確認してください。モデルをインチへ切り替えても、明示的に「mm」「N」などと表示された欄はその単位で入力します。

設定・操作	1.1.4 での入力・表示
通常のモデル座標・長さ・寸法	モデルの入力・表示単位
面積・体積	モデル単位の 2 乗・3 乗
用紙サイズ・紙面上の配置寸法	mm
寸法スタイルの文字・矢印寸法	mm。モデルの実寸と区別する
全体設定の文字高さ・グリッド既定値	mm 表記の欄は mm
線幅・線種・表示設定	欄に表示された mm または px
穴コマンドの径・深さ	mm 表記に従う
解析の形状寸法・変位	mm。荷重や物性は各欄の単位
IFC 接続口の位置・径・幅・高さ	mm。方向ベクトルは長さではない

解析で特に混同しやすい単位

荷重は N、合計荷重は N または kgf、分布荷重は N/mm²、線荷重は N/mm です。ヤング係数と応力の N/mm²は MPa と同じ大きさです。温度は℃、熱流束は W/m²、熱伝達率は W/(m²K)です。

解析入力を inch・lbf・psi へ切り替える機能はありません。インチ表示のモデルを使う場合も、解析画面に示された mm・SI 系の値へ合わせます。幾何形状は取り込み時の実寸を使います。

確認の順番

単位を変更した後は、既知の長さを一つ計測し、寸法表示を確認し、最後にレイアウトの印刷プレビューを確認します。長さ表示だけの変更を理由に、形状へ 25.4 倍などの尺度変更を重ねないでください。

28 部品・外部形式の単位

ブロックマネージャーで使う JDC 部品は mm 基準です。挿入先のモデル表示がインチでも、実寸を保って配置します。外部形式では、ファイルの単位情報と図面の表示単位を区別してください。

形式・対象	読み込み／挿入	書き出し
JDC 部品	mm 基準の実寸を保持	編集用原本として JDC を保存
DWG／DXF	ファイルの単位情報で換算。未指定時は全体設定の読み込み単位	mm 座標・mm ヘッダー。ブロック定義の基準単位は別途保持
単位を持つ外部ブロック	定義の基準単位を保持し、配置時に換算	定義単位と配置倍率の関係を保持
STEP／SAT	ファイルの単位情報から換算	使用する保存形式の設定と受け取り側を確認
STL／OBJ	全体設定の「ファイルの長さ単位」を指定	mm で出力

STL・OBJ を開く前に

通常はファイル自体に長さ単位がありません。［全体設定］→［STL / OBJ 読み込み］→［ファイルの長さ単位］を出力元に合わせてから開きます。既定は mm です。出力元がメートルならメートルを選びます。

実寸が合わない場合

1. 出力元で既知の寸法と、保存時に選んだ単位を確認します。

2. 読み込み後に同じ箇所を計測します。25.4 倍、1,000 倍などの差があれば単位指定を確認します。

3. 単位不明のファイルは正しい読み込み単位で開き直します。モデルの表示単位を変えるだけでは、誤って取り込まれた実寸は直りません。

ダイナミックブロックの定義単位とルックアップ式は保持します。外部の変更済みブロックを初期化して編集する場合は、現在の外観が初期状態へ戻るため、先に JDC の原本を残します。

リボンと表示モード

現在の画面を見ながら、使用するタブを選びます。API サンプルは付属サンプルを配置した場合に表示されます。カスタマイズの管理グループ名は「拡張機能」です。

タブ	主な用途
ホーム	作画、モデリング、修正、モデル修正、注釈
挿入	ブロックの作成・挿入、ダイナミックブロック、書き出し
管理／計測	図面や表示の管理、形状の計測
解析	2D／3D 解析。STD／Pro
板金	板金設定、作成、編集、展開・再曲げ、展開図
API サンプル／カスタマイズ	サンプルコマンド、DLL の読込状況と配置先

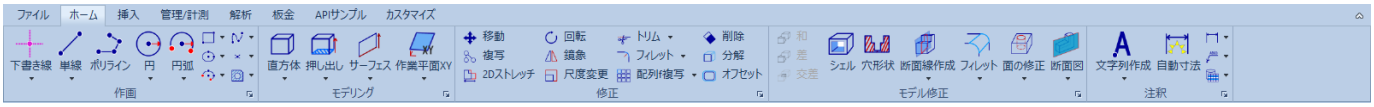


図 1　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。

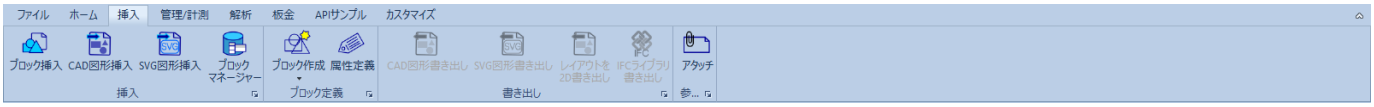


図 2　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。



図 3　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。



図 4　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。



図 5　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。



図 6　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。



図 7　現行リボンの実画面。画面幅によりグループの折りたたみ方が変わります。

文字の見え方と隠線表示

文字の塗りつぶしは照明の影響を受けず、等角表示でも指定色で表示します。モデルの面の照明は維持します。表示モードは既存のシェーディング／シェーディング（エッジ無）を残し、シェーディング（エッジ無・隠線）を追加しています。

モード	見え方
シェーディング	面を塗り、モデルのエッジも表示
シェーディング（エッジ無）	モデルのエッジを消し、2D 図形を前面に表示
シェーディング（エッジ無・隠線）	モデルのエッジを消し、面の後ろに隠れる 2D 図形は表示しない

モデル表面を線分で表現する場合は、隠線の追加モードを選ぶと、裏側の線が前面へ透けて出るのを防げます。画面上に常に注記を見せたい場合は従来モードを使います。

ショートカットと印刷時の確認

ショートカットを登録する

管理／計測タブの管理グループから「ショートカット登録」を開きます。検索欄でコマンド名または ID を絞り、割り当てるコマンドを選びます。

キーと Ctrl／Shift／Alt を指定します。「キーを押して指定」では、実際のキーまたはマウスのサイドボタンを入力できます。「登録／変更」を押し、既存の割り当てがある場合は置き換え内容を確認します。

「OK」で保存・適用します。「キャンセル」または右上の×では、その画面での変更を破棄します。登録解除・初期設定への復帰も「OK」で確定します。設定は JSON 形式で読み込み・書き出しできます。

F1〜F24、英字、数字、テンキー、矢印、Delete、XButton1／XButton2 などに対応します。Esc、Enter、Space、Tab、Alt+F4 などの予約キーは登録できません。文字・数値・表の入力中や別のダイアログでは入力操作を優先します。

割り当ては図面ではなく利用者ごとの設定です。日本語／英語を切り替えても同じコマンドへの割り当てを保持します。別プロセスで開いているウィンドウには、次回起動時に読み込まれます。

大判用紙の印刷プレビュー

印刷画面で用紙、向き、縮尺を選び「プレビューを更新」します。A1・A0 なども、使用するプリンターが実際に対応する用紙と向きを確認してください。

ドライバーが指定用紙を別のサイズへ置き換える場合は、指定用紙のプレビューを表示して不一致を案内します。その状態では印刷できません。対応するプリンターまたは用紙に切り替え、再度プレビューを更新します。

固定縮尺では図面が用紙からはみ出すことがあります。全体を収めるには「用紙に合わせる」または適切な縮尺を選びます。プレビュー対応によってプリンターの物理的な用紙対応範囲が広がることはありません。

画面フィットと複雑な図形の表示

表示したい画層だけを表示状態にして「画面フィット」を実行します。非表示の画層上にある 2D 図形などは、表示範囲の算出対象から除外します。形状が見つからない場合は、画層の表示、選択フィルター、対象モデルを確認してください。

複雑な 3D 図形では、初めて開くとき、エンジン更新後、または図面内容を変更したときに、表示形状の再計算が必要です。同じ図面を開き直す際には表示キャッシュを利用します。表示の準備にかかる時間は形状と端末によって異なります。

移動・コピーでは基点と移動先を指定し、プレビューを確認して確定します。入力中のプレビューは表示用の変換で更新し、確定時に実形状を処理します。確定後の位置とコピー数を確認し、必要なら「元に戻す」を使います。

図面を開く際は、保存された表示情報とキャッシュを利用して表示準備を行います。モデルとレイアウトの切り替えでは有効な表示情報を再利用します。図面変更直後やキャッシュがない場合には再計算が必要です。

画面フィットは現在表示されている図形を対象にします。モデルだけをフィットしたい場合は不要な 2D レイヤを非表示にしてから実行します。MCP による表示切り替えは view.activate、既存図面の読み込みは document.open を使います。

ソリッドの体積を計算する

ソリッドを選択し、プロパティ欄の「Volume」の文字をダブルクリックします。同じ行のメニューから「体積を計算」を選ぶ方法もあります。「計算中」が数値へ変わるまで待ちます。

複雑な形状を選ぶたびに重い計算を始めないよう、体積は明示操作で計算します。未計算の表示だけで、ソリッドの異常を意味するわけではありません。計算中もほかの操作を行えます。

計算値は数値近似です。mm の図面では mm^3となり、モデルの表示単位に応じて体積の換算を行います。選択し直した場合は計算値を保持します。形状を変更した後や、図面を開き直した後は再計算してください。

第 2 部 リボンコマンドリファレンス

コマンド索引

コマンド名の右端は本文のページ番号です。同じ名称が複数ある場合は、タブとグループ、本文のコマンド ID を併せて確認してください。

ファイル　ファイル操作

新規作成 57
開く 58
名前を付けて保存 60
上書き保存 61
ファイルを閉じる 62
印刷 63

ホーム　作画

ポリライン 64
下書き線　クロス 65
下書き線　水平 66
下書き線　垂直 67
下書き線　２点角度 68
下書き線　角度 69
下書き線　参照角度 70
下書き線　点と接物 71
下書き線　接物と接物 72
下書き線　基準固定 参照要素 73
下書き線　基準移動 参照要素 74
下書き線　すべて削除 75
下書き線　表示・非表示 76
線分　単 線 77
線分　連続線 78
線分　長さと角度 79
線分　端点と接物 80
線分　接物と接物 81
線分　接物と角度 82
線分　中点垂直 83
線分　角度 2 等分 84
線分　平行線 距離 85
線分　平行線 点 86
線分　垂線 点と要素 87
線分　マルチライン 88
線分 円の中心線 89
線分 2 線間の中心線 90

ホーム　作画

円　中心と点 91
円　中心と半径 92
円　2 点 93
円　3 点 94
円　2 点と半径 95
円　3 接物 96
円　2 接物と点 97
円　2 接物と半径 98
円　接物と点と半径 99
円 接物・中心軌跡・半径 100
円弧　中心と点 101
円弧　中心と半径 102
円弧　2 点と半径 103
円弧　3 点 104
円弧　3 接物 105
円弧　2 接物と点 106
円弧　2 接物と半径 107
円弧　接物と点と半径 108
矩形　対角矩形 109
矩形　中心と幅高さ 110
矩形　幅と高さと角度 111
矩形　2 点と高さ 112
平行四辺形 113
ポリゴン 114
楕円　3 点 115
楕円　2 点と 2 方向 116
楕円弧　3 点 117
楕円弧　2 点と 2 方向 118
スプライン 119
螺旋 122
点 123
ディバイド 124
ハッチング 125
単純ハッチング 126
境界線 127
閉図形チェック 128
イメージ 129
イメージ 2 点指示 130

ホーム　モデリング

ソリッド　直方体 131

ソリッド　円柱 132
ソリッド　円錐 133
ソリッド　球 134
円環体 135
押し出し 136
回転 137
ロフト 138
パスロフト 139
押し出し 140
回転 141
平面サーフェース 142
ロフト 143
パスロフト 144
ネットワーク 145
作業平面 XY 146
作業平面 XY オフセット 147
作業平面 XZ 148
作業平面 XZ オフセット 149
作業平面 YZ 150
作業平面 YZ オフセット 151
作業平面オフセット 152
モデル上の面 153
面とエッジと角度 154
パスと曲線平面 155
作業平面 3 点 156
作業平面パスに垂直 157
WCS X 角度 158
WCS Y 角度 159
WCS Z 角度 160
UCS X 角度 161
UCS Y 角度 162
UCS Z 角度 163

ホーム　修正

移動 164
複写 165
2D ストレッチ 166
回転 167
鏡象 168
尺度変更 169
削除 170
分解 171

オフセット 172
トリム 173
サーフェストリム 174
分割 175
結合 176
重なり要素を除去 177
延長 178
包絡 179
クリップ円形 180
クリップ矩形 181
フィレット 182
全角フィレット 183
面取り 184
全角面取り 185
3D フィレット 186
3D 面取り 187
矩形配列複写 188
円形配列複写 189

ホーム　モデル修正

和 190
差 191
交差 192
シェル 193
穴形状 基本形状と入口 194
穴形状 連続クリックと配列配置 195
穴形状 ねじ情報と簡略表示 196
穴形状 ねじ深さの自動計算 197
穴形状 ダブルクリック編集 199
断面線 200
ソリッド-面切断 201
ソリッド-干渉チェック 202
フィレット 204
面取り 207
フィレット削除 208
フィレットサイズ変更 209
面の除去 210
面の移動 211
面の回転 214
面の複写 215
面のオフセット 216
面のテーパー 217

曲線を面へ投影 218
断面平面を作成 220
ライブ断面の ON/OFF 221
断面線を XY 平面に 222
断面線設定 223

ホーム　注釈

自動寸法 224
文字入力 225
文字入力 2 点指示 226
文字輪郭パス化 227
マルチテキスト 228
水平 229
垂直 230
長さ 231
要素間 232
半径 233
直径 234
要素間角度 235
3 点角度 236
4 点角度 237
面取り寸法 238
引き出し線 239
面にラベル貼付け　240
座標引き出し線 241
バルーン 242
マーカー 243
幾何公差 244
データム 245
仕上げ記号 246
部品表 247

挿入　挿入

ブロック挿入 248
CAD 図形挿入 249
SVG 図形挿入 250
ブロック 251

挿入　ブロック定義

属性定義 252
ブロック作成 253
ブロック書き出し 254

挿入　書き出し

CAD 図形書き出し 255
SVG 図形書き出し 256
レイアウトを 2D 書き出し 257

挿入　参照

アタッチ 258

管理/計測　管理

名前削除 259
参照管理 260

管理/計測　計測

２点間距離 261
UCS 角度 262
面の角度 263
閉図形の周長 264
面積・断面特性 265
面と面の距離 266
ソリッド容積・重心 267

レイアウト 右クリックコマンド

レイアウト 表示とビューの管理 268
レイアウト VIEWBASE ベースビュー 269
レイアウト 投影ビューを追加 270
レイアウト 面に正対する補助投影ビュー 271
レイアウト 円形の詳細図を追加 272
レイアウト 断面ビュー 273
レイアウト 稜線を非表示と再表示 274
レイアウト 稜線の太さを変更 275

タブメニュー：　ファイル	グループ：ファイル操作
 コマンド名：　新規作成	コマンド ID：File.New

新しい図面を作成します。

操作方法：

① 起動したばかりの場合　　実行するとファイルタブからホームタブに切り替わります

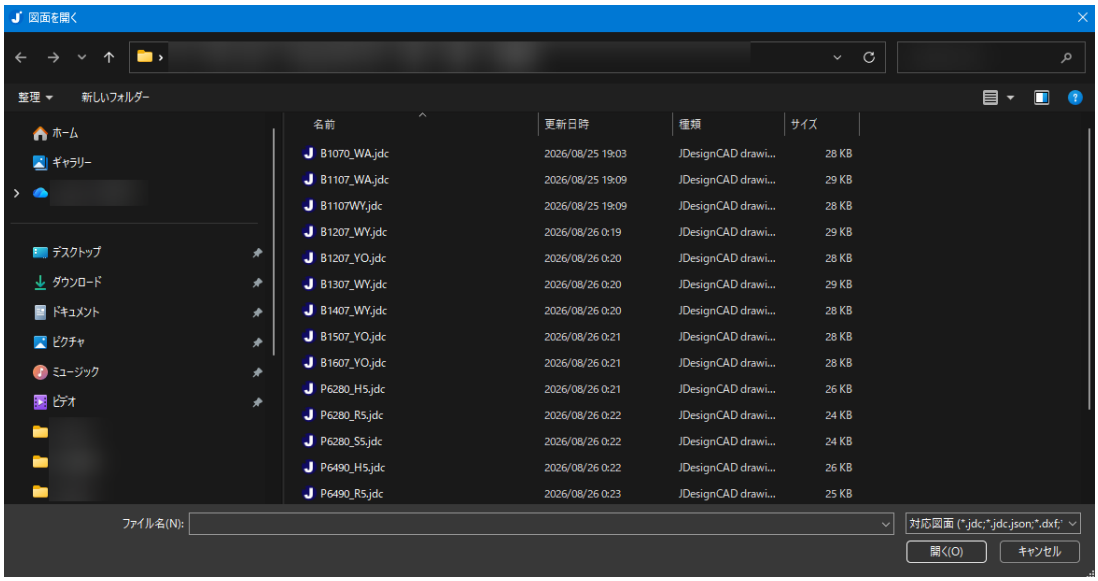
② 既に開いている図面がある場合　JDesignCAD をもう一つ起動して新規作成を行います

タブメニュー：　ファイル	グループ：ファイル操作
 コマンド名：　開く	コマンド ID：File.Open

JDC、DXF、DWG、JWW、STEP、STL などの図面／モデルを開きます。

操作方法：

開くダイアログが表示されます



③ 起動したばかりの場合　　実行するとファイルタブからホームタブに切り替わりデータを開きます

④ 既に開いている図面がある場合　JDesignCAD をもう一つ起動してデータを開きます

SXF（P21／SFC）の読み書き

SXF は土木図面などの 2D 交換に使います。読み込みは Free でも利用でき、保存・書き出しは STD・Pro で利用できます。書き出しは SXF 3.1 です。

操作手順

1. ［開く］または CAD 図形の挿入で、SXF P21（*.p21）または SXF SFC（*.sfc）を選びます。

2. 既知寸法、画層、線種、文字、寸法の配置を確認します。黒色は背景に合わせて白黒を切り替えて表示します。

3. 書き出す場合は、アクティブモデルの XY 平面上の 2D 図形を用意し、［名前を付けて保存］または書き出しから P21／SFC を選びます。

4. 元の編集情報を残すため、JDC も保存します。SXF 保存後も元の図面で編集を続けられます。

要素	交換時の扱い
通常の長さ・角度・弧長・半径・直径寸法	測定対象と配置を再現できれば編集可能な寸法
特殊な引出線・公差など	寸法として再現できない場合は線・文字へ分解
ハッチ	対応範囲で保持。未対応模様は線へ展開、または輪郭と警告
マルチテキスト	行の配置を保った単行文字として出力
塗りつぶし	中抜きを含む平面領域として変換
線幅 0	書き出し時に 0.01 mm。元図面の線幅は保持

対応範囲

3D 図形や傾いた面を自動で XY へ投影する機能ではありません。全 SXF 要素の完全互換や OCF 認証を示すものでもありません。警告は確認し、相手側でも寸法・文字・尺度を確認します。

以前の版で線・文字へ分解済みの寸法は、元の測定点を自動復元できません。編集可能な元の JDC から現行版で出力し直してください。

タブメニュー：　ファイル	グループ：ファイル操作
 コマンド名：　名前を付けて保存	コマンド ID：File.Save.As

現在の図面へ名前と保存形式を指定して保存します。

操作方法：
現在開いているファイルを名前を付けて保存します

Free 版は JDC ファイルのみ登録できます

STD・Pro 版は JDC,DWG,DXF,JWW で保存できます
ただし、DWG,DXF は 3DSOLID データはメッシュデータになります
3DSOLID データを AUTOCAD に持っていきたい場合は CAD データ出力コマンドから STEP で出力して AutoCAD の Stepin コマンドで読み込んでください

JWW は 2 D 図形のみです

タブメニュー：　ファイル	グループ：ファイル操作
 コマンド名：　上書き保存	コマンド ID：File.Save

現在の図面を既存ファイルへ上書き保存します。

操作方法：

ファイルを一度でも保存している場合は、そのまま上書きされます

ファイルをまだ保存していなかった場合は、名前を付けて保存コマンドが実行されます

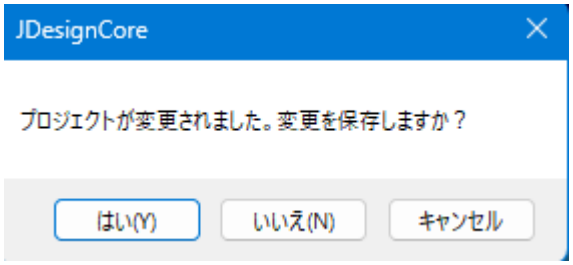
JDC 以外を読み込んだ場合も名前を付けて保存コマンドが実行されます

タブメニュー：　ファイル	グループ：ファイル操作
コマンド名：ファイルを閉じる	コマンド ID：(MenuId なし：内部処理)

保存確認後、現在の図面を閉じてスタート画面へ戻ります。

操作方法：

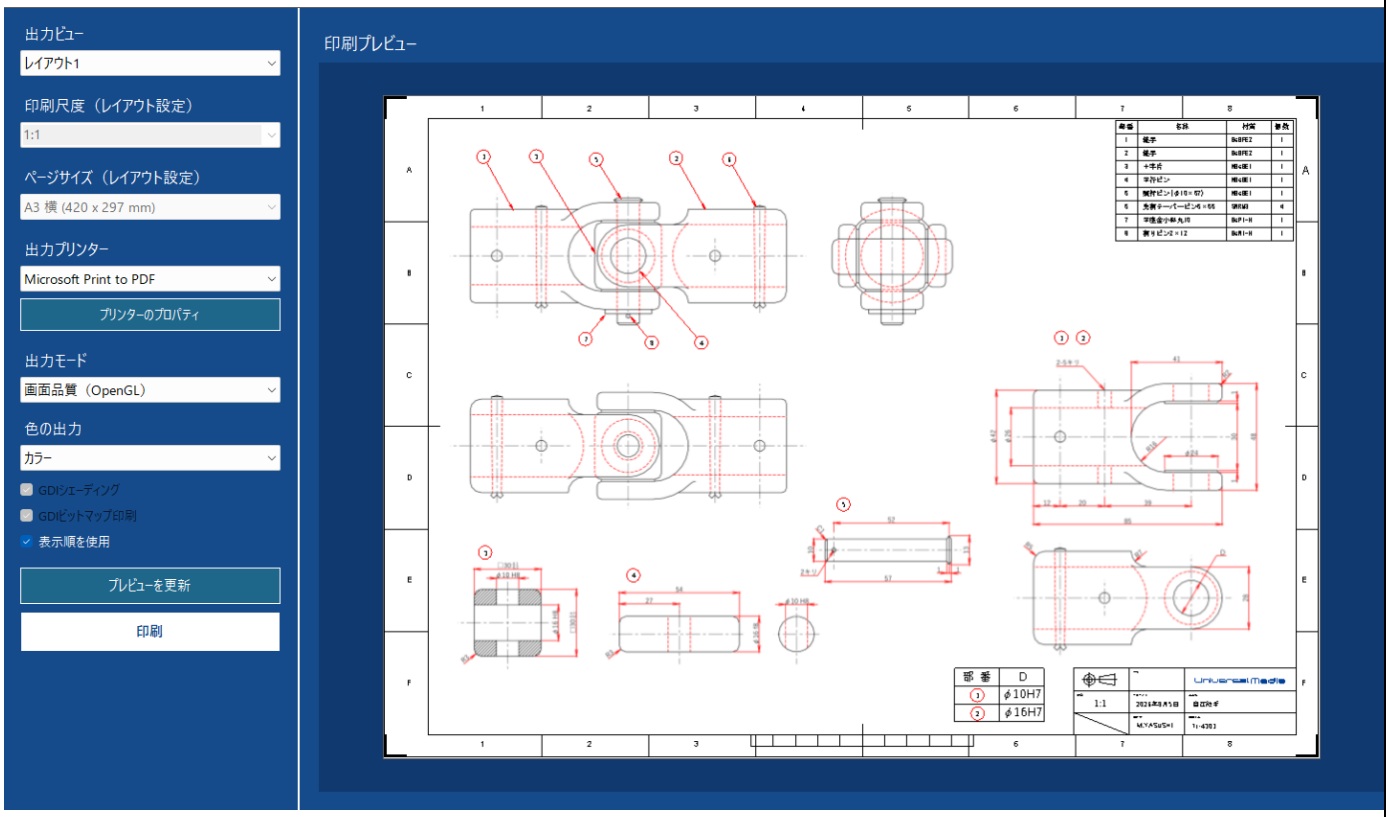
実行すると現在開いているファイルが閉じます、修正しているのに一度も保存していない場合は下記ダイアログが出てきます



タブメニュー：　ファイル	グループ：ファイル操作
コマンド名：印刷	コマンド ID：(MenuId なし：印刷画面)

印刷ビュー、尺度、用紙、プリンター、出力モードを設定してプレビュー／印刷します。

操作方法：



■ 出力ビュー：モデルビューやレイアウトビューなど出力対象のビューを選択します
■ 印刷尺度：モデルビューが出力対象の場合、出力尺度を設定します、
　レイアウトビューの場合は 1：1 固定です
■ ページサイズ：モデルビューが出力対象の場合、用紙サイズを設定します、
　レイアウトビューの場合はレイアウト側で決められた用紙サイズになります
■ 出力プリンター：プリンターを設定します、プリンターのプロパティボタンでプリンタードライバの
　設定ができます
■ 出力モード：出力画像に品質を設定します、OpenGL と GDI での出力を設定できます

■ 色の出力：出力する色を設定します、カラー、モノクロで出力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：　ポリライン	コマンド ID：Constr.Polyline

連続指示したポイントにポリラインを作成します

操作方法

1　始点をマウスの左ボタンで指示します。

2　次の頂点を順に指示します。入力した点が線分でつながります。

3　終点まで入力したら、開いた形状か閉じた形状かを確認して実行します。

プロパティエクスプローラー：

■ 点列：各頂点の位置を指定します。

■ 長さ：次の線分の長さを指定できます。

■ 終端：閉じるを選ぶと、終点と始点を結びます。

操作のポイント：

連続線とは異なり、つながった線を 1 つのポリラインとして扱います。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　クロス	コマンド ID：Constr.DraftLine.Cross

現在の作業平面上で、指定点を通る水平・垂直の下書き線を作成します。

操作手順：

メッセージ

① 下書き線を通す点を指示してください

　下書き線を入力する基準点をマウスの左ボタンでヒットします

プロパティエクスプローラー：

■通過点

　X,Y,Z 座標を入力できます

タブメニュー：　ホーム	グループ：作画
コマンド名：下書き線　水平	コマンド ID：Constr.DraftLine.Horizontal

現在の作業平面上で、指定点を通る水平の下書き線を作成します。

操作手順：

メッセージ

① 下書き線を通す点を指示してください
　下書き線を入力する基準点をマウスの左ボタンでヒットします

プロパティエクスプローラー：

■通過点
　X,Y,Z 座標を入力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　垂直	コマンド ID：Constr.DraftLine.Vertical

現在の作業平面上で、指定点を通る垂直の下書き線を作成します。

操作手順：

メッセージ

① 下書き線を通す点を指示してください

　下書き線を入力する基準点をマウスの左ボタンでヒットします

プロパティエクスプローラー：

■通過点

　X,Y,Z 座標を入力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　2 点角度	コマンド ID：Constr.DraftLine.TwoPointAngle

1 点目と 2 点目で角度を指定し、その方向の下書き線を作成します。

操作手順：

メッセージ

① 角度下書き線 1 点目を指示してください
　角度下書き線を入力する 1 点目をマウスの左ボタンでヒットします

② 角度を決める 2 点目を指示してください
　角度下書き線を決定する 2 点目を入力します

プロパティエクスプローラー：

■1 点目
　X,Y,Z 座標を入力できます

■2 点目
　X,Y,Z 座標を入力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　角度	コマンド ID：Constr.DraftLine.Angle

UCS 基準の角度を指定し、指定点を通る下書き線を連続作成します。

操作手順：

メッセージ

① UCS 基準の角度を入力してください

プロパティエクスプローラーから角度を入力します

② 下書き線を通す点を指示してください

下書き線を挿入する点を指示します

プロパティエクスプローラー：

■ 角度

角度を入力します

■ 通過点

X,Y,Z で入力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　参照角度	コマンド ID：Constr.DraftLine.RelativeAngle

基準要素と相対角度を指定し、指定点を通る下書き線を連続作成します。

操作手順：

メッセージ

① 相対角度の基準にする要素を指示してください
　相対角度の基準線を選択します

② 選択した要素からの相対角度を入力してください
　プロパティエクスプローラーで角度を入力します

③ 下書き線を通す点を指示してください
　下書き線を挿入する点を指示します

プロパティエクスプローラー：

■ 角度
　角度を入力します

■ 通過点
　X,Y,Z で入力できます

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　点と接物	コマンド ID：Constr.DraftLine.PointTangent

指定点から選択要素へ接する下書き線を連続作成します。

操作手順：

1　下書き線を通す始点を指示します。

2　円や円弧など、接線を引く要素を選択します。

3　目的の接線が表示される側を確認して確定します。続けて別の接線を作成できます。

プロパティエクスプローラー：

■ 始点：接線下書き線が通る点です。座標でも指定できます。

■ 接物：下書き線を接する要素を選択します。

操作のポイント：

円の内側など、指定点から接線を引けない位置では作成できません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　接物と接物	コマンド ID：Constr.DraftLine.TwoTangents

2 つの選択要素に接する下書き線を連続作成します。

操作手順：

1　1 つ目の接物を選択します。

2　2 つ目の接物を選択します。

3　2 つの要素に接する候補を確認して確定します。候補が違うときは指示位置を変えます。

プロパティエクスプローラー：

■ 接物：1 つ目の接線対象です。

■ 2 つ目の接物：2 つ目の接線対象です。

操作のポイント：

接物の組合せによっては、外側と内側など複数の接線候補があります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　基準固定 参照要素	コマンド ID：Constr.DraftLine.FixedReferenceParallel

基準線を選択し、入力した距離ごとに平行な下書き線を連続作成します。

操作手順：

1　平行線の基準にする要素を選択します。

2　基準線からの距離を入力し、作成する側を確認して確定します。

3　続けて別の距離を入力すると、最初の基準線からその距離の下書き線を作成します。

プロパティエクスプローラー：

■ 基準線：平行下書き線の基準となる要素です。

■ 距離：固定した基準線からの距離です。

操作のポイント：

例えば 10、20 と入力すると、最初の基準線から 10 と 20 の位置に作成します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　基準移動 参照要素	コマンド ID：Constr.DraftLine.MovingReferenceParallel

基準線を選択し、入力した距離で平行な下書き線を作成します。作成した下書き線が次の基準になります。

操作手順：

1　平行線の基準にする要素を選択します。

2　距離を入力して下書き線を作成します。

3　次の距離を入力します。直前に作成した下書き線が新しい基準になります。

プロパティエクスプローラー：

■ 基準線：初回の基準となる要素です。

■ 距離：現在の基準線から次の下書き線までの距離です。

操作のポイント：

同じ側へ 10、20 と入力すると、最初の基準線から 10 と 30 の位置になります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　すべて削除	コマンド ID：Constr.DraftLine.DeleteAll

描かれている下書き線をすべて削除します。

操作手順：

1　残しておきたい下書き線がないか確認します。

2　コマンドを実行すると、下書き線をまとめて削除します。

操作のポイント：

一時的に画面から消したいだけの場合は、「表示・非表示」を使用してください。削除と非表示は異なります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：下書き線　表示・非表示	コマンド ID：Constr.DraftLine.ToggleVisible

描かれている下書き線を表示・非表示します。

操作手順：

1　コマンドを実行すると、下書き線の表示状態が切り替わります。

2　再度実行すると、元の表示状態へ戻ります。

操作のポイント：

非表示にしても下書き線のデータは残ります。作図後に下書き線だけを隠して確認するときに使用します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　単 線	コマンド ID：Constr.Line.TwoPoints

線分: 始点と終点を指定します

操作手順：

1　始点を左クリックします。座標値で指定することもできます。

2　マウスを移動して終点を指示すると、2 点を結ぶ線分が作成されます。

プロパティエクスプローラー：

■ 始点・終点：線分の両端の座標を指定します。

■ 長さ・方向：線分を数値で指定するときに使用します。

操作のポイント：

作業平面とスナップ先を確認してから点を指示してください。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　連続線	コマンド ID：Constr.Line.RPoint

連続線コマンドを実行します。

操作手順：

1　最初の点を指示します。

2　次の点を指示して線分を作成します。終点が次の線分の始点になります。

3　必要な線分を続けて入力し、終了するときは Esc キーで入力を終えます。

プロパティエクスプローラー：

■ 点列：線分をつなぐ点を順に指定します。

■ 長さ：次に作成する線分の長さを指定できます。

操作のポイント：

各区間を線分として作成します。全体を 1 要素として扱う場合はポリラインを使用します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　長さと角度	コマンド ID：Constr.Line.PointLengthAngle

線分: 始点、長さ、方向を指定します

操作手順：

1　線分の始点を指示します。

2　長さと方向を入力します。

3　プレビューで向きを確認して確定します。

プロパティエクスプローラー：

■ 始点：線分の基準となる点です。

■ 長さ：作成する線分の長さを入力します。

■ 方向：作業平面上での線分の方向を指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　端点と接物	コマンド ID：Constr.Line.PointTangent

線分: 始点と接線要素を指定します

操作手順：

1　線分の始点を指示します。

2　円や円弧など、接する要素を選択します。

3　作成したい接線に近い位置を指示し、プレビューを確認して確定します。

プロパティエクスプローラー：

■ 始点：接線の一方の端点です。

■ 接線対象：もう一方の端が接する要素です。

操作のポイント：

接点は幾何条件から求められます。円の内側の点からは、その円への接線を作成できません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　接物と接物	コマンド ID：Constr.Line.TwoTangents

線分: 2 つの接線要素を指定します

操作手順：

1　1 つ目の接線対象を選択します。

2　2 つ目の接線対象を選択します。

3　目的の接線候補を確認して確定します。

プロパティエクスプローラー：

■ 接線対象：1 つ目の接物です。

■ 第 2 接線対象：2 つ目の接物です。

操作のポイント：

指示位置によって選ばれる候補が変わる場合があります。接したい側の近くを指示します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　接物と角度	コマンド ID：Constr.Line.TangentAngle

線分: 方向と接線要素を指定します

操作手順：

1　接線対象の円や円弧などを選択します。

2　線分の方向と長さを入力します。

3　接点を線分のどの位置に置くか確認して確定します。

プロパティエクスプローラー：

■ 接線対象：接線を求める要素です。

■ 方向・長さ：接線の向きと長さを指定します。

■ 位置：接点を始点側または中央に置く設定です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　中点垂直	コマンド ID：Constr.Line.MiddlePerp

線分: 要素の垂直二等分線を作図します

操作手順：

1　垂直二等分線の基準となる線分を選択します。

2　作成する線分の長さを入力します。

3　基準線分の中点を通る向きを確認して確定します。

プロパティエクスプローラー：

■ 基準要素：二等分する線分です。

■ 長さ：作成する垂直線分の長さです。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　角度 2 等分	コマンド ID：Constr.Line.BisectAngle

線分: 角の二等分線を作図します

操作手順：

1　角度の基準となる 1 本目の線を選択します。

2　2 本目の線を選択します。

3　二等分する側と線分の長さを確認して確定します。

プロパティエクスプローラー：

■ 角の二等分線の要素：角度を決める 2 つの要素を指定します。

■ 長さ：二等分線の長さを入力します。

操作のポイント：

2 本の線が作るどちらの角を二等分するか、プレビューで確認してください。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　平行線 距離	コマンド ID：Constr.Line.ParallelDist

線分: 指定線分から指定距離の平行線を作図します

操作手順：

1　平行に複写する基準線を選択します。

2　距離を入力し、作成する側を指示します。

3　基準線との間隔を確認して確定します。

プロパティエクスプローラー：

■ 基準線：平行線の元になる線分です。

■ 距離：基準線からの垂直距離です。

操作のポイント：

斜めの線でも、間隔は水平・垂直方向の距離ではなく基準線からの距離になります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　平行線 点	コマンド ID：Constr.Line.ParallelPoint

線分: 指定線分に平行で点を通る線を作図します

操作手順：

1　平行線の基準となる線分を選択します。

2　新しい線分を通したい点を指示します。

3　通過点と平行方向を確認して確定します。

プロパティエクスプローラー：

■ 基準線：方向と長さの基準となる線分です。

■ 点：平行線を通す点を指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　垂線 点と要素	コマンド ID：Constr.Line.PointPerp

線分: 別の要素に垂直な線を作図します

操作手順：

1　垂線を引き始める点を指示します。

2　垂線の基準となる要素を選択します。

3　垂足と線分を確認して確定します。

プロパティエクスプローラー：

■ 始点：垂線の始点です。

■ 基準要素：垂線を下ろす線や曲線です。

操作のポイント：

曲線では、目的の垂足に近い位置を指示して候補を確認します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：線分　マルチライン	コマンド ID：Constr.MultiLine

中心点列から 2 重から 4 重の連続平行線を作成します

操作手順：

1　平行線の本数と、中心経路からの各オフセットを設定します。

2　中心経路の始点と次の点を順に指示します。

3　必要なら閉じる設定を選び、形状を確認して実行します。

プロパティエクスプローラー：

■ 本数：2〜4 本の範囲で指定します。

■ オフセット：中心経路に対する各線の位置を設定します。

■ 閉じる：始点と終点をつないだ形状にする場合に選択します。

操作場所： ホーム	グループ：作画
線分 円の中心線	コマンド ID：Constr.Line.CenterMark

概要

円・円弧の中心を通る十字の中心線を作成します。2D 作図、3D モデルの円形稜線、レイアウトの投影図で使用できます。

操作手順

1　「ホーム」→「単線」の▼→「円の中心線」を選びます。

2　対象の円または円弧をクリックします。3D では円形稜線、レイアウトでは目的のビュー内の円を選びます。

3　必要に応じて「はみ出し長さ」を設定し、チェックマークで確定します。

入力と作成結果

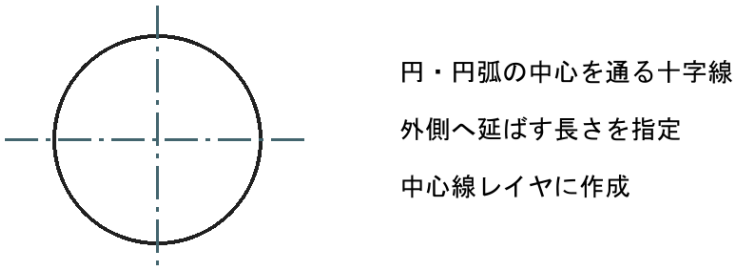
はみ出し長さは円の外側へ延ばす長さです。モデルではモデル上の長さ、レイアウトでは用紙上の mm で指定します。

操作のポイント

中心線は「中心線」レイヤへ、色・線種・線幅を ByLayer で作成します。レイヤがなければ新規登録し、既存レイヤがあればその設定を使用します。

関連付け

3D 稜線とレイアウトの投影図から作った中心線は参照元に関連付けられます。独立した 2D 円から作った中心線は通常の線で、円の変更には自動追従しません。参照元を大きく変更した場合は位置を確認してください。



操作イメージ

操作場所： ホーム	グループ：作画
線分 2 線間の中心線	コマンド ID：Constr.Line.CenterBetween

概要

平行な 2 本の線の中間に中心線を作成します。任意の 2 点ではなく、2 本の直線要素を指定するコマンドです。

操作手順

1　「ホーム」→「単線」の▼→「2 線間の中心線」を選びます。

2　1 本目の直線をクリックし、続いて平行な 2 本目の直線をクリックします。

3　はみ出し長さを設定し、中央に作成される線を確認してチェックマークで確定します。

対象の選び方

2D では独立した線分、3D ではモデルの直線稜線を選びます。2D の線と 3D 稜線を混在させて指定することはできません。レイアウトでは同じビュー内の 2 本を指定します。

入力と作成結果

中心線は 2 本の線の中間を通り、指定したはみ出し長さだけ端を延長します。レイアウトのはみ出し長さは紙面上の mm です。

操作のポイント

平行でない線や同じ線を 2 回選んだ場合は作成できません。「中心線」レイヤと関連付けの扱いは「円の中心線」と共通です。



操作イメージ

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　中心と点	コマンド ID：Constr.Circle.CenterPoint

円: 中心点と指示点

操作手順：

1　円の中心を指示します。

2　円周上に置きたい点へマウスを移動します。

3　半径を確認し、その点をクリックして円を作成します。

プロパティエクスプローラー：

■ 中心：円の中心座標です。

■ 円周上の点：中心からこの点までの距離が半径になります。

■ 半径：円の大きさを確認できます。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　中心と半径	コマンド ID：Constr.Circle.CenterRadius

円:中心と半径

操作手順：

1　円の中心を指示します。

2　半径を入力します。直径入力を使う場合は直径欄に入力します。

3　中心と大きさを確認して確定します。

プロパティエクスプローラー：

■ 中心：円の中心座標です。

■ 半径：円の中心から円周までの距離です。

■ 直径：半径の 2 倍の値です。半径と直径を取り違えないでください。

操作のポイント：

半径 10 の円は直径 20 になります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　2 点	コマンド ID：Constr.Circle.2Points

円:2 点を指示して円を作画します

操作手順：

1　直径の一方の端になる点を指示します。

2　反対側の点を指示します。

3　2 点を直径の両端とする円が作成されます。

プロパティエクスプローラー：

■ 円上の第 1 点・第 2 点：直径の両端を指定します。

■ 中心・半径：2 点から求まった値を確認できます。

操作のポイント：

中心は 2 点の中点、半径は 2 点間距離の半分になります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　3 点	コマンド ID：Constr.Circle.3Points

円:3 点を指示して円を作画します

操作手順：

1　円周上の 1 点目と 2 点目を指示します。

2　円周上の 3 点目を指示します。

3　3 点を通る円を確認して確定します。

プロパティエクスプローラー：

■ 円上の第 1 点～第 3 点：円周が通る 3 点です。

■ 中心・半径：3 点から求まった値を確認できます。

操作のポイント：

3 点が一直線上にある場合や、同じ点を重ねた場合は円が定まりません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　2 点と半径	コマンド ID：Constr.Circle.TwoPointsRadius

円:2 点と半径を指示して円を作画します

操作手順：

1　円が通る 1 点目と 2 点目を指示します。

2　半径を入力します。

3　2 点に対してどちら側に中心を置くか、候補を確認して確定します。

プロパティエクスプローラー：

■ 円上の第 1 点・第 2 点：円周が通る 2 点です。

■ 半径・直径：どちらかの値で大きさを指定します。

操作のポイント：

半径は 2 点間距離の半分以上が必要です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　3 接物	コマンド ID：Constr.Circle.ThreeTangents

円: 3 つの要素に接する円を作図します

操作手順：

1　1 つ目、2 つ目、3 つ目の接線対象を順に選択します。

2　接したい側の近くを指示して候補を確認します。

3　3 つの要素すべてに接する円を確定します。

プロパティエクスプローラー：

■ 第 1～第 3 接線対象：円が接する要素です。

■ 半径：条件から求められた大きさを確認します。

操作のポイント：

先に半径を決めるコマンドではありません。半径は 3 つの接線条件で決まります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　2 接物と点	コマンド ID：Constr.Circle.TwoTangentsPoint

円: 2 つの要素に接し、円周点を指定します

操作手順：

1　1 つ目と 2 つ目の接線対象を選択します。

2　作成する円の円周が通る点を指示します。

3　2 つの要素に接し、指定点を通る候補を確認して確定します。

プロパティエクスプローラー：

■ 第 1・第 2 接線対象：円が接する 2 要素です。

■ 円上の点：中心ではなく円周上の通過点です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　2 接物と半径	コマンド ID：Constr.Circle.TwoTangentsRadius

円: 2 つの要素に接し、半径を指定します

操作手順：

1　1 つ目と 2 つ目の接線対象を選択します。

2　半径または直径を入力します。

3　希望する側の円が表示されることを確認して確定します。

プロパティエクスプローラー：

■ 第 1・第 2 接線対象：円が接する要素です。

■ 半径・直径：作成する円の大きさです。

操作のポイント：

指定した半径で両方の要素に接する円が存在しない場合は、条件を変更してください。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円　接物と点と半径	コマンド ID：Constr.Circle.TangentPointRadius

円: 1 つの要素に接し、点と半径を指定します

操作手順：

1　円を接する要素を選択します。

2　円周が通る点を指示します。

3　半径を入力し、目的の候補を確認して確定します。

プロパティエクスプローラー：

■ 第 1 接線対象：円が接する要素です。

■ 円上の点：円周の通過点です。

■ 半径・直径：円の大きさを指定します。

操作のポイント：

「円上の点」は中心点ではありません。

操作場所： ホーム	グループ：作画
 **円 接物・中心軌跡・半径**	コマンド ID：Constr.Circle.TangentCenterCurveRadius

概要

指定した要素に接し、中心が別の要素上にある円を、指定半径で作成します。「中心軌跡」は、円の中心を置ける線・円・円弧などです。

操作手順

1　「ホーム」→「円」の▼→「接物・中心軌跡・半径」を選びます。

2　作成する円を接触させる要素を選びます。

3　円の中心を置く要素を、中心軌跡として選びます。

4　半径を入力して Enter を押します。

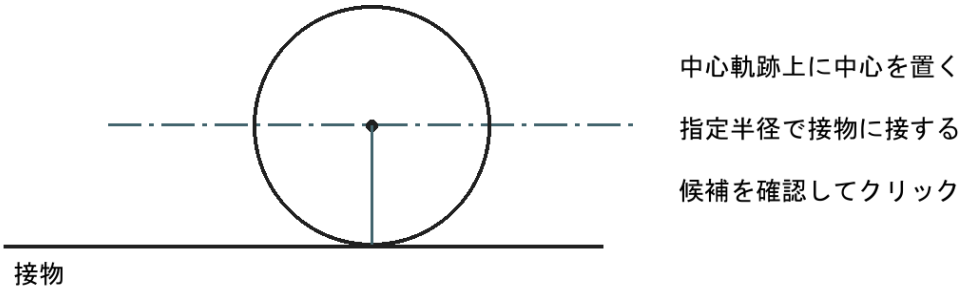
5　マウスを目的の円の近くへ動かし、候補を確認してクリックします。

入力項目

接物：円を接触させる要素。中心軌跡：中心を置く要素。半径：作成する円の大きさ。複数の候補がある場合はマウス位置で選び分けます。

操作のポイント

接物と中心軌跡は同一平面上の要素を指定します。接点と中心は選択要素の実際の範囲内に限られ、延長部分には作成しません。条件を満たす円がないときは、半径または対象を見直してください。元の要素は変更されません。



操作イメージ

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　中心と点	コマンド ID：Constr.Arc.CenterStartEnd

円弧: 中心、始点、終点を指定します

操作手順：

1　円弧の中心を指示します。

2　円弧の始点を指示して半径を決めます。

3　終点側の方向を指示し、回る向きを確認して確定します。

プロパティエクスプローラー：

■ 中心・始点・終点：円弧の位置と範囲を指定します。

■ 方向：円弧が回る向きを設定します。

■ 半径・開始角・終了角：大きさと角度範囲を確認・指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　中心と半径	コマンド ID：Constr.Arc.CenterRadius

円弧: 中心、半径、開始角、終了角を指定します

操作手順：

1　円弧の中心と半径を指定します。

2　開始角と終了角を入力します。

3　回転方向を確認して円弧を確定します。

プロパティエクスプローラー：

■ 中心：円弧の中心です。

■ 半径・直径：円弧の大きさを指定します。

■ 開始角・終了角・方向：円弧を残す範囲を決めます。

操作のポイント：

短い側と長い側を取り違えないよう、プレビューで円弧全体を確認します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　2 点と半径	コマンド ID：Constr.Arc.2PointsRadius

円弧: 2 点と半径を指定します

操作手順：

1　円弧の 1 点目と 2 点目を指示します。

2　半径を入力します。

3　膨らむ側を確認して確定します。

プロパティエクスプローラー：

■ 円周上の第 1 点・第 2 点：円弧の両端を指定します。

■ 半径・直径：円弧の大きさを指定します。

■ 中心：選んだ候補の中心を確認できます。

操作のポイント：

半径が 2 点間距離の半分より小さいと円弧は作成できません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　3 点	コマンド ID：Constr.Arc.3Points

円弧: 始点、中間点、終点を指定します

操作手順：

1　円弧の始点を指示します。

2　途中を通る点を指示します。

3　終点を指示して、3 点を通る円弧を作成します。

プロパティエクスプローラー：

■ 円周上の第 1 点～第 3 点：始点、中間点、終点の順に指定します。

■ 中心・半径：3 点から決まる値を確認します。

操作のポイント：

中間点をどちら側に置くかで、円弧の膨らむ向きが変わります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　3 接物	コマンド ID：Constr.Arc.ThreeTangents

円弧: 3 つの要素に接する円弧を作図します

操作手順：

1　3 つの接線対象を順に選択します。

2　接する側と、円弧として作成する範囲をプレビューで確認します。

3　目的の候補を確定します。

プロパティエクスプローラー：

■ 第 1～第 3 接線対象：円弧が接する要素です。

■ 半径：接線条件から決まる大きさを確認します。

操作のポイント：

同じ接物でも選択位置によって候補が異なる場合があります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　2 接物と点	コマンド ID：Constr.Arc.TwoTangentsPoint

円弧: 2 つの要素に接し、円周点を指定します

操作手順：

1　2 つの接線対象を選択します。

2　円弧が通る円周上の点を指示します。

3　接する側と円弧の範囲を確認して確定します。

プロパティエクスプローラー：

■ 第 1・第 2 接線対象：円弧が接する要素です。

■ 円周上の点：円弧が通る位置を指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　2 接物と半径	コマンド ID：Constr.Arc.TwoTangentsRadius

円弧: 2 つの要素に接し、半径を指定します

操作手順：

1　2 つの接線対象を選択します。

2　半径または直径を入力します。

3　接点と円弧の向きを確認して確定します。

プロパティエクスプローラー：

■ 第 1・第 2 接線対象：円弧が接する要素です。

■ 半径・直径：円弧の大きさを指定します。

操作のポイント：

元の線を角丸に加工したい場合は、修正の「フィレット」も使用できます。

タブメニュー：　ホーム	グループ：作画
 コマンド名：円弧　接物と点と半径	コマンド ID：Constr.Arc.TangentPointRadius

円弧: 1 つの要素に接し、点と半径を指定します

操作手順：

1　円弧を接する要素を選択します。

2　円弧が通る点を指示します。

3　半径を入力し、候補を確認して確定します。

プロパティエクスプローラー：

■ 第 1 接線対象：接する要素です。

■ 円周上の点：円弧の通過点です。

■ 半径・直径：大きさを指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：矩形　対角矩形	コマンド ID：Constr.Rect.2DiagonalPoints

矩形: 対角 2 点を指定します

操作手順：

1　矩形の 1 つ目の角を指示します。

2　反対側の角へマウスを移動し、対角点を指示します。

プロパティエクスプローラー：

■ 始点：矩形の一方の角です。

■ 対角点：始点に対する反対側の角です。

操作のポイント：

矩形の辺は現在の作業平面の軸方向にそろいます。傾いた矩形は角度指定を使用します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：矩形　中心と幅高さ	コマンド ID：Constr.Rect.CenterWidthHeight

矩形: 中心点、幅、高さを指定します

操作手順：

1　矩形の中心点を指示します。

2　サイズ点を指示するか、幅と高さを入力します。

3　中心から両側へ振り分けられた形状を確認して確定します。

プロパティエクスプローラー：

■ 中心点：矩形の中心です。

■ サイズ点：マウスで幅と高さを決めます。

■ 幅・高さ：矩形全体の寸法です。

操作のポイント：

幅 40、高さ 20 なら、中心から左右 20、上下 10 の範囲になります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：矩形　幅と高さと角度	コマンド ID：Constr.Rect.PointWidthHeightAngle

矩形: 点、幅、高さ、方向を指定します

操作手順：

1　矩形の始点を指示します。

2　幅と高さを入力します。

3　基準辺の方向を指定し、向きを確認して確定します。

プロパティエクスプローラー：

■ 始点：矩形の基準となる角です。

■ 幅・高さ：2 辺の寸法です。

■ 方向：幅方向の辺の角度を指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：矩形　2 点と高さ	コマンド ID：Constr.Rect.2PointsHeight

矩形: 基準 2 点と高さを指定します

操作手順：

1　基準となる辺の始点を指示します。

2　同じ辺の第 2 点を指示します。

3　高さを入力し、矩形を作る側を確認して確定します。

プロパティエクスプローラー：

■ 始点・第 2 点：基準辺の位置、長さ、方向を決めます。

■ 高さ：基準辺に垂直な方向の寸法です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　平行四辺形	コマンド ID：Constr.Rect.Parallelogram

平行四辺形: 3 点を指定します

操作手順：

1　平行四辺形の始点を指示します。

2　主方向となる隣の頂点を指示します。

3　副方向を指定し、2 つの辺で囲まれる形状を確認して確定します。

プロパティエクスプローラー：

■ 始点：2 辺の共通の角です。

■ 主方向・副方向：隣り合う辺の向きと長さを指定します。

操作のポイント：

2 方向が平行になると面積を持つ形状になりません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　ポリゴン	コマンド ID：Constr.Polygon

内接、外接多角形を作成します。

操作手順：

1　辺数と作成方法を設定します。

2　多角形の中心と半径を指定します。

3　基準角度を指定して確定します。

プロパティエクスプローラー：

■ 辺数：多角形の辺の数です。

■ 作成方法：内接または外接を選択します。

■ 中心点・半径・角度：位置、大きさ、向きを指定します。

操作のポイント：

内接は頂点が基準円上、外接は各辺が基準円に接する形状です。同じ半径でも大きさが異なります。

タブメニュー：　ホーム	グループ：作画
 コマンド名：楕円　3 点	コマンド ID：Constr.Ellipse.CenterRadius

楕円: 中心と半径を指定します

操作手順：

1　楕円の中心を指示します。

2　第 1 頂点を指示して長軸側を決めます。

3　第 2 頂点を指示して短軸側を決め、確定します。

プロパティエクスプローラー：

■ 中心：楕円の中心座標です。

■ 第 1 頂点・第 2 頂点：2 つの軸の頂点です。

■ 第 1 半径・第 2 半径・楕円角度：軸の半径と傾きを指定できます。

操作のポイント：

半径欄は全幅ではなく、中心から頂点までの値です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：楕円　2 点と 2 方向	コマンド ID：Constr.Ellipse.2PointsDirections

楕円:2 点を指示して 2 方向で作成します

操作手順：

1　楕円上の第 1 点と、その点での接線方向を指定します。

2　第 2 点と、その点での接線方向を指定します。

3　2 点と 2 方向を満たす楕円を確認して確定します。

プロパティエクスプローラー：

■ 第 1 点・第 2 点：楕円周上の点です。

■ 第 1 方向・第 2 方向：各点での接線方向です。

操作のポイント：

方向は楕円の長軸・短軸そのものではなく、指定点での接線です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：楕円弧　3 点	コマンド ID：Constr.Ellipsearc.CenterRadius

楕円弧: 中心と 2 つの半径を指定します

操作手順：

1　中心、第 1 頂点、第 2 頂点を指定して楕円の形を決めます。

2　開始角と終了角を指定します。

3　残す円弧の方向を確認して確定します。

プロパティエクスプローラー：

■ 中心・第 1 半径・第 2 半径：楕円の位置と大きさです。

■ 楕円角度：楕円全体の傾きです。

■ 開始角・終了角・方向：楕円弧の範囲を指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：楕円弧　2 点と 2 方向	コマンド ID：Constr.Ellipsearc.2PointsDirections

楕円弧:2 点を指示して 2 方向で作成します

操作手順：

1　第 1 点と、その点での接線方向を指定します。

2　第 2 点と、その点での接線方向を指定します。

3　2 点の間を結ぶ楕円弧の方向を確認して確定します。

プロパティエクスプローラー：

■ 第 1 点・第 2 点：楕円弧の端となる点です。

■ 第 1 方向・第 2 方向：各点の接線方向です。

■ 方向：作成する楕円弧の回る向きを指定します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　スプライン	コマンド ID：Constr.BSpline.Points

NURBS スプラインを作成します

操作手順：

1　スプラインが通る点を順に指示します。

2　必要な点を追加し、曲線の形を確認します。

3　開いた曲線か閉じた曲線かを設定し、実行します。

プロパティエクスプローラー：

■ 点列：曲線を形作る点を追加・確認します。

■ モード：開く／閉じるを選択します。

操作のポイント：

点を過剰に増やすと曲線が波打つ場合があります。形状を表すために必要な点を入力します。

クロソイドを作図する

［ホーム］→作画グループの［スプライン］の▼から［クロソイド］を選びます。任意の作業平面に配置できる平面曲線です。

基本式と入力の意味

$A^2 = R \times L$。A はクロソイドパラメーター、L は曲率ゼロの KA から曲線に沿った距離、R はその位置の曲率半径です。A・L・R には同じ長さ単位を使います。

1. 作業平面を確認し、配置方法、数値指定方法、左回り／右回りを選びます。

2. 基点と方向を示す点を指定します。

3. A と開始・終了距離、または A と開始・終了半径を入力します。

4. プレビューの長さ・向きを確認して、チェックで確定します。

入力例	設定値	得られる曲線
KA から全区間を作る	A＝100、開始距離 0、終了距離 100	長さ 100、終点半径 100
途中の区間を作る	A＝100、開始距離 20、終了距離 100	長さ 80、始点半径 500、終点半径 100
半径指定の 0	開始半径 0	無限大の半径、すなわち直線への接続点

例の値はモデルが mm なら mm です。メートル表示なら同じ数値はメートルの長さを意味するため、設計寸法に合わせます。

配置時の注意

［基点と進入方向］では KA と接線方向を指定します。［始点と終点方向］の 2 点目も方向指定で、その点へ届く長さを自動計算する指定ではありません。長さは A と距離／半径で決まります。

作成後はプロパティで KA の位置、A、開始・終了距離を変更できます。成立しない値は採用されず、元の値を保持します。

クロソイド基本型と横断線

IP からクロソイド―円弧―クロソイドを作る

［スプライン］の▼→［クロソイド基本型（IP 指定）］を実行します。進入側の点、接線交点 IP、退出側の点を順に指定し、A と終了半径 R を入力します。左右に同じ A を使う対称な基本型です。

3 点は進入・退出方向を決めます。既存の直線は自動で切り詰めません。交角に対してクロソイドが長すぎる場合は、A を小さくするか R を大きくして、成立する線形にします。同一直線、180 度の折り返し、左右で異なる A の基本型は対象外です。

垂線・接線・等間隔の横断線

［クロソイドの垂線・接線・横断線］でモードを選び、対象曲線をクリックします。ここでの距離は「表示している曲線の始点からの弧長」です。KA からの距離とは区別します。

モード	指定する値
距離指定の垂線	始点からの距離、左右の長さ
距離指定の接線	始点からの距離、前後の長さ
外部点からの垂線	外部の点。複数解では最短の垂線
等間隔の横断線	開始距離、間隔、左右の長さ

長さ 100 の曲線に開始 0・間隔 20・左右各 5 を指定すると、幅 10 の横断線を 0、20、40、60、80、100 の 6 か所に作ります。終点が間隔の整数倍でなければ、終点の線は追加しません。1 回の上限は 2,000 本です。

保存と再編集

JDC ではクロソイド専用要素として保存します。XY 平面の通常形状は SXF のクロソイド、DWG／DXF はポリライン近似になります。作った横断線は独立した線分で、曲線の変更へ自動追従しません。縦断線形・片勾配・道路断面の自動生成は行いません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　螺旋	コマンド ID：Constr.Helix

らせん状のパスラインを作成します

操作手順：

1　らせんの中心を指示します。

2　下半径、上半径、高さ、巻数を入力します。

3　回転方向と形状を確認して確定します。

プロパティエクスプローラー：

■ 下半径・上半径：始端側と終端側の半径です。同じ値なら一定径になります。

■ 高さ・巻数：軸方向の高さと回転数です。

■ 1 巻の滑らかさ：補間点数です。大きくするとデータ量も増えます。

■ 回転方向：作業平面の法線方向を基準に指定します。

操作のポイント：

らせんのパスを作るコマンドです。線材などの立体化は、このパスを使って別途行います。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　点	コマンド ID：Constr.Point

単純な点を作図します

操作手順：

1　点を作成する位置を左クリックします。

2　必要な位置へ続けて点を配置し、終了するときは入力を終了します。

プロパティエクスプローラー：

■ 点：X、Y、Z 座標で位置を指定できます。

操作のポイント：

画面上の見え方は点の表示設定に従います。円や穴を作るコマンドではありません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　ディバイド	コマンド ID：Constr.Divide

選択した要素を指定数で等分し、分割位置に点を作成します。

操作手順：

1　等分する線分・円弧・曲線を選択します。

2　分割数を入力します。

3　等分位置の点を確認して確定します。

プロパティエクスプローラー：

■ 要素：等分位置を求める曲線です。

■ 分割数：曲線を何等分するかを指定します。

操作のポイント：

元の曲線を切断する操作ではなく、等分位置に点を作成する操作です。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　ハッチング	コマンド ID：Constr.Hatch.WithHoles

中抜き形状があるハッチングを作成します

操作手順：

1　閉じた外周と、穴になる閉じた輪郭を用意します。

2　ハッチングする領域の内側を指示します。

3　穴の部分が除かれていることとハッチングスタイルを確認して確定します。

プロパティエクスプローラー：

■ 内点：塗る領域の内部を指示します。

■ ハッチングスタイル：線の模様などを設定します。

操作のポイント：

輪郭に隙間があると領域を検出できません。「閉図形チェック」で端点を確認してください。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　単純ハッチング	コマンド ID：Constr.Hatch.Inside

一番外側の境界線を選んでハッチング作成します

操作手順：

1　ハッチングする閉じた領域を用意します。

2　その内側を指示します。

3　塗られる範囲とスタイルを確認して確定します。

プロパティエクスプローラー：

■ 内点：ハッチング領域を指定します。

■ ハッチングスタイル：表示する模様を設定します。

操作のポイント：

中抜きする輪郭がある場合は、通常の「ハッチング」を使って穴の扱いを確認してください。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　境界線	コマンド ID：Constr.Boundary

内側点を指定して、閉じた境界をパスラインとして作成します。

操作手順：

1　線や円弧で囲まれた領域を用意します。

2　境界線を作る領域の内側点を指示します。

3　検出された輪郭を確認して確定します。

プロパティエクスプローラー：

■ 内側点：閉領域の内部を指定します。

操作のポイント：

新しい閉じたパスラインを作成します。元の線を直接つなぎ直したい場合は「結合」を使用します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　閉図形チェック	コマンド ID：JDesign.CheckClosedContours

選択した線・円弧・ポリライン・パス・スプライン等が閉図形を構成するか確認し、開いた端点をプレビュー表示します。

操作手順：

1　確認する線・円弧・ポリラインなどを選択します。

2　コマンドを実行して判定結果を確認します。

3　開いた端点が表示された場合は、その位置を拡大して修正します。

操作のポイント：

チェックだけでは隙間は修正されません。端点を合わせてから再度確認し、押し出しやハッチングに使用します。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　イメージ	コマンド ID：Constr.Picture.RefPointWidthHeight

イメージ: 固定点、幅、高さを指定します

操作手順：

1　配置する画像ファイルを選択します。

2　画像の基準位置を指示します。

3　幅、高さ、角度を設定して確定します。

プロパティエクスプローラー：

■ 画像・位置：使用するファイルと配置位置を指定します。

■ 幅・高さ・基準線角度：大きさと傾きを指定します。

■ 縦横比の保持：画像を変形させたくない場合に使用します。

操作のポイント：

画像は線や円などの CAD 要素には変換されません。

タブメニュー：　ホーム	グループ：作画
 コマンド名：　イメージ 2 点指示	コマンド ID：Constr.Picture.RefPoint2Directions

イメージ: 固定点、幅方向、高さ方向を指定します

操作手順：

1　画像ファイルを選択し、基準位置を指示します。

2　幅方向と高さ方向を画面上で指定します。

3　大きさと傾きを確認して確定します。

プロパティエクスプローラー：

■ 位置：画像の基準点です。

■ 基準線方向・左辺方向：画像を配置する 2 方向です。

■ 幅・高さ・拡大縮小：配置サイズを指定します。

■ 縦横比の保持・矩形：画像の形状を保つための設定です。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：ソリッド　直方体	コマンド ID：Constr.Box

長さ、幅、高さ、または任意方向でボックスを作成します

操作手順：

1　直方体の始点を指示します。

2　長さ、幅、高さを入力します。任意方向に作る場合は各軸方向も指定します。

3　挿入モードと形状を確認し、実行します。

プロパティエクスプローラー：

■ 始点：直方体の基準となる角です。

■ 長さ・幅・高さ：3 方向の寸法です。

■ X・Y・Z 方向：直方体の各辺の向きを指定します。

■ 挿入モード：新規、和演算、差演算などを選択します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：ソリッド　円柱	コマンド ID：Constr.Cylinder

軸と半径、または半径と高さで円柱を作成します

操作手順：

1　底面の中心点を指示します。

2　半径と高さを指定します。傾いた円柱は軸上の点で方向と高さを指定します。

3　挿入モードを確認して実行します。

プロパティエクスプローラー：

■ 中心点：円柱の基準位置です。

■ 軸上の点：円柱軸の方向と長さを決めます。

■ 半径・高さ：円柱の大きさです。

■ 挿入モード：既存ソリッドへの追加・切削などを設定します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：ソリッド　円錐	コマンド ID：Constr.Cone

軸と半径、または半径と高さで円錐を作成します

操作手順：

1　底面の中心点を指示します。

2　第 1 半径、第 2 半径、高さを入力します。必要なら軸上の点を指定します。

3　形状と挿入モードを確認して実行します。

プロパティエクスプローラー：

■ 中心点・軸上の点：円錐の位置と軸を指定します。

■ 1. 半径・2. 半径：両端の半径です。第 2 半径は 0 にもできます。

■ 高さ：軸方向の長さです。

■ 挿入モード：新規作成や既存ソリッドへの演算を選択します。

操作のポイント：

第 2 半径を 0 にすると先端が尖った円錐、0 以外にすると円錐台になります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：ソリッド　球	コマンド ID：Constr.Sphere

中心と半径で球を作成します

操作手順：

1　球の中心点を指示します。

2　半径を入力します。

3　挿入モードを確認して実行します。

プロパティエクスプローラー：

■ 中心点：球の中心座標です。

■ 半径：中心から球面までの距離です。

■ 挿入モード：新規作成、和演算、差演算などを指定します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　円環体	コマンド ID：Constr.Torus

中心、主半径、副半径でトーラスを作成します。必要に応じて軸上の点を指定できます。

操作手順：

1　円環体の中心点を指示します。

2　主半径と副半径を入力します。

3　必要なら軸方向を指定し、形状と挿入モードを確認して実行します。

プロパティエクスプローラー：

■ 中心点・軸上の点：円環体の位置と軸方向です。

■ 主半径：中心から管の断面中心までの距離です。

■ 副半径：管の断面の半径です。

■ 挿入モード：既存ソリッドとの演算方法です。

操作のポイント：

一般的なリング形状では、外半径は主半径＋副半径、内半径は主半径－副半径になります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　押し出し	コマンド ID：Constr.Solid.FaceExtrude

面または閉じた曲線を押し出してソリッドを作成します

操作手順：

1　閉じた断面または面を選択します。穴付き領域は「内点」からも指定できます。

2　押し出しモードと振り分けを選び、高さまたは到達先の平面を指定します。

3　挿入モードを新規・和演算・差演算などから選び、方向を確認して実行します。

プロパティエクスプローラー：

■ 高さ・振り分け：中心振り分けでは入力した高さを両側へ半分ずつ配分します。

■ 指定平面オブジェクト：面まで押し出す場合の到達先です。

■ テーパー角度：正の値は先端側を小さく、負の値は大きくします。

■ 押し出し平面・方向：断面の基準平面や押し出す向きを指定します。

■ オフセット距離・方向：押し出し開始位置をずらします。

■ 押し出しパス：曲線に沿って押し出す場合に指定します。

■ 挿入モード：和は結合、差は既存ソリッドからの切削です。

操作のポイント：

例：中心振り分けで高さ 140 を入力すると、断面から両側 70 ずつ押し出します。差演算では切削形状が対象に届くことを確認してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　回転	コマンド ID：Constr.Solid.FaceRotate

面または閉じた曲線を回転してソリッドを作成します

操作手順：

1　回転させる閉じた断面または面を選択します。

2　回転軸を選択します。軸を数値で指定する場合は軸上の点と軸方向を入力します。

3　角度、回転方向、振り分けを設定します。

4　挿入モードとプレビューを確認して実行します。

プロパティエクスプローラー：

■ 軸・軸上の点・軸方向：回転中心となる直線を決めます。

■ 角度・回転方向：回す範囲と向きを指定します。

■ 回転モード：角度指定または指定平面までを選択します。

■ 振り分け・オフセット：回転範囲の配分と開始位置を指定します。

■ 挿入モード：新規・和演算・差演算などを指定します。

操作のポイント：

差演算では回転体と対象ソリッドが交差する必要があります。断面だけでなく軸の入力も確定してから実行してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　ロフト	コマンド ID：Constr.Solid.Loft

複数の閉じた断面を選択順に通るロフトソリッドを作成します

操作手順：

1　閉じた平面断面を、つなぐ順番に 2 個以上選択します。

2　必要な断面がそろったら、挿入モードを設定します。

3　ねじれや断面の順番をプレビューで確認し、Enter キーで確定します。

プロパティエクスプローラー：

■ 断面：選択した順がロフトの接続順になります。

■ 挿入モード：新規・和演算・差演算を指定します。

操作のポイント：

選択済みの断面を再クリックすると選択から外れます。ソリッドには閉じた断面を使用してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　パスロフト	コマンド ID：Constr.Solid.PathLoft

複数の閉じた断面を中心パスに沿ってロフトします

操作手順：

1　閉じた平面断面を、パスに沿う順番で 2 個以上選択します。

2　Enter キーで断面選択を確定し、開いた中心パスを選択します。

3　挿入モードと形状を確認し、Enter キーで実行します。

プロパティエクスプローラー：

■ 断面：途中形状を含む断面列を指定します。

■ 中心パス：断面の位置と姿勢が沿う曲線です。

■ 挿入モード：新規・和演算・差演算を選択します。

操作のポイント：

事前選択から実行する場合は、断面を順に選び、中心パスを最後に選択します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　押し出し	コマンド ID：Constr.Face.ObjectExtrude

要素を押し出して面を作成します。要素にはパス、曲線、接続された曲線を使用できます。

操作手順：

1　押し出す線・円弧・パスなどを選択します。

2　高さと振り分け、または到達先の平面を指定します。

3　方向と面の形を確認して実行します。

プロパティエクスプローラー：

■ 要素：押し出して面にする曲線です。

■ 高さ・振り分け：押し出す長さと両側への配分です。

■ 指定平面オブジェクト：面まで押し出す場合に指定します。

■ 押し出し平面・方向・オフセット：基準と開始位置を設定します。

■ 押し出しパス：曲線に沿わせる場合に指定します。

操作のポイント：

サーフェスを作成するコマンドです。体積を持つ立体が必要な場合はソリッドの押し出しを使用します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　回転	コマンド ID：Constr.Face.ObjectRotate

要素を回転して面を作成します。要素にはパス、曲線、接続された曲線を使用できます。

操作手順：

1　回転させる曲線またはパスを選択します。

2　回転軸を選択するか、軸上の点と軸方向を指定します。

3　角度と方向を設定し、プレビューを確認して実行します。

プロパティエクスプローラー：

■ 要素：回転させる曲線です。

■ 軸・軸上の点・軸方向：回転軸を指定します。

■ 角度・方向・振り分け：回転する範囲を指定します。

■ 指定平面・オフセット：到達先や開始角度を指定する場合に使用します。

操作のポイント：

作成されるのは回転サーフェスです。ソリッドの回転とは区別してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　平面サーフェース	コマンド ID：Constr.Face.FromObject

単独の曲線、接続された曲線、またはパスを平面の面に変換します

操作手順：

1　同じ平面上にある閉じた曲線またはつながった輪郭を指定します。

2　穴付き領域を作る場合は内点を指示します。

3　面になる範囲を確認して確定します。

プロパティエクスプローラー：

■ 曲線：面の外周となる閉じた輪郭です。

■ 内点：穴を含む領域を内部の点で指定します。

操作のポイント：

厚みは付きません。厚みのある形状は、この面を押し出して作成します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　ロフト	コマンド ID：Constr.Face.Loft

複数断面を選択順に通るロフトサーフェスを作成します

操作手順：

1　断面を接続する順に 2 個以上選択します。

2　面のねじれや接続方向を確認します。

3　Enter キーでロフトサーフェスを確定します。

プロパティエクスプローラー：

■ 断面：開断面同士、または閉断面同士を指定します。

操作のポイント：

開いた断面と閉じた断面を混在させないでください。閉断面を使っても、このコマンドはサーフェス作成です。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　パスロフト	コマンド ID：Constr.Face.PathLoft

複数断面を中心パスに沿ってロフトします

操作手順：

1　断面をパスに沿う順番で 2 個以上選択します。

2　Enter キーで断面選択を終え、開いた中心パスを選択します。

3　パスに沿って変化する面を確認し、Enter キーで確定します。

プロパティエクスプローラー：

■ 断面：開断面同士、または閉断面同士を指定します。

■ 中心パス：断面の位置と姿勢を決める曲線です。

操作のポイント：

中心パスは 1 本に連続した曲線を用意します。断面の選択順を確認してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　ネットワーク	コマンド ID：Constr.Face.NetworkSurface

交差する 2 方向の曲線群からサーフェスを作成します

操作手順：

1　1 方向目の曲線群を 2 本以上選択します。

2　Enter キーで 1 方向目を確定します。

3　交差する 2 方向目の曲線群を 2 本以上選択します。

4　プレビューを確認し、Enter キーで確定します。

プロパティエクスプローラー：

■ 第 1 方向の曲線群：面の一方向を形作る曲線です。

■ 第 2 方向の曲線群：第 1 方向と交差する曲線です。

操作のポイント：

2 方向の曲線が格子状に交差するよう用意します。交差していない曲線を見かけだけで重ねても面は作成できません。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 XY	コマンド ID：DrawingPlane.StandardXY

X/Y 平面を現在の作業平面に設定します

操作手順：

1　コマンドを実行して、ワールド座標の XY 平面を作業平面に設定します。

2　UCS の軸表示と作業平面の向きを確認してから作図します。

操作のポイント：

既存のモデルを回転する操作ではありません。これから作図するときの基準平面を変更します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 XY オフセット	コマンド ID：DrawingPlane.StandardXY.Offset

オフセット付きの X/Y 平面を現在の作業平面に設定します

操作手順：

1　XY 平面を基準にしたオフセットコマンドを実行します。

2　法線オフセットに距離を入力します。

3　平面が移動する側を確認して確定します。

プロパティエクスプローラー：

■ 法線オフセット：基準平面から法線方向への符号付き距離です。

操作のポイント：

正負で移動する側が変わります。画面の上下方向と一致するとは限りません。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 XZ	コマンド ID：DrawingPlane.StandardXZ

X/Z 平面を現在の作業平面に設定します

操作手順：

1　コマンドを実行して、ワールド座標の XZ 平面を作業平面に設定します。

2　UCS の軸表示と作業平面の向きを確認してから作図します。

操作のポイント：

既存のモデルを回転する操作ではありません。これから作図するときの基準平面を変更します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 XZ オフセット	コマンド ID：DrawingPlane.StandardXZ.Offset

オフセット付きの X/Z 平面を現在の作業平面に設定します

操作手順：

1　XZ 平面を基準にしたオフセットコマンドを実行します。

2　法線オフセットに距離を入力します。

3　平面が移動する側を確認して確定します。

プロパティエクスプローラー：

■ 法線オフセット：基準平面から法線方向への符号付き距離です。

操作のポイント：

正負で移動する側が変わります。画面の上下方向と一致するとは限りません。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 YZ	コマンド ID：DrawingPlane.StandardYZ

Y/Z 平面を現在の作業平面に設定します

操作手順：

1　コマンドを実行して、ワールド座標の YZ 平面を作業平面に設定します。

2　UCS の軸表示と作業平面の向きを確認してから作図します。

操作のポイント：

既存のモデルを回転する操作ではありません。これから作図するときの基準平面を変更します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 YZ オフセット	コマンド ID：DrawingPlane.StandardYZ.Offset

オフセット付きの Y/Z 平面を現在の作業平面に設定します

操作手順：

1　YZ 平面を基準にしたオフセットコマンドを実行します。

2　法線オフセットに距離を入力します。

3　平面が移動する側を確認して確定します。

プロパティエクスプローラー：

■ 法線オフセット：基準平面から法線方向への符号付き距離です。

操作のポイント：

正負で移動する側が変わります。画面の上下方向と一致するとは限りません。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面オフセット	コマンド ID：DrawingPlane.Current.Offset

現在の作業平面を法線方向へ指定距離だけ移動します

操作手順：

1　現在の作業平面を確認します。

2　法線オフセットに移動量を入力します。

3　移動後の作業平面を確認して確定します。

プロパティエクスプローラー：

■ 法線オフセット：現在の作業平面からの符号付き距離です。

操作のポイント：

平面内の X・Y 方向には動かず、平面に垂直な方向へ平行移動します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　モデル上の面	コマンド ID：DrawingPlane.Tangential

モデル上の平面フェイスを選択します

操作手順：

1　基準にしたいモデルの平面フェイスを選択します。

2　選択面に沿った作業平面と UCS の向きを確認します。

3　その平面上で作図を始めます。

操作のポイント：

平面を選択する操作です。曲面を同じ平面として扱うことはできません。面を選びにくい場合は視点を変えてください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　面とエッジと角度	コマンド ID：DrawingPlane.AngleToFace

平面フェイス上の直線エッジを軸に、指定角度の作業平面を作成します。

操作手順：

1　基準となる平面フェイスを選択します。

2　その面上の直線エッジを回転軸として選択します。

3　角度を入力し、傾いた作業平面を確認して確定します。

プロパティエクスプローラー：

■ 面：傾ける前の基準平面です。

■ エッジ：平面を回す軸となる直線です。

■ 角度：基準面からの傾きを指定します。

操作のポイント：

円弧エッジではなく、対象面上の直線エッジを指定します。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　パスと曲線平面	コマンド ID：DrawingPlane.OfCurve

パスまたは平面曲線が存在する平面を作業平面にします

操作手順：

1　平面上にある円・円弧・パスなどを選択します。

2　その曲線が存在する平面を確認し、作業平面として使用します。

操作のポイント：

空間的にねじれた曲線は、1 つの平面を定義できません。平面曲線を指定してください。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面 3 点	コマンド ID：DrawingPlane.Three.Points

作業平面を 3 点指示して定義

操作手順：

1　作業平面の原点になる 1 点目を指示します。

2　X 方向を決める 2 点目を指示します。

3　平面の向きを決める 3 点目を指示し、UCS を確認して確定します。

プロパティエクスプローラー：

■ 3 点：原点、軸方向、平面の向きを順に決めます。

操作のポイント：

3 点が一直線上にあると平面が定まりません。指示する順番で平面の向きが変わります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　作業平面パスに垂直	コマンド ID：DrawingPlane.Path.Perpendicular

パスまたは曲線の指定端点に、接線と垂直な作業平面を作成します

操作手順：

1　基準にするパスまたは曲線を選択します。

2　作業平面を置く側の端点を指定します。

3　端点での接線に垂直な平面になっていることを確認します。

操作のポイント：

パスに沿って押し出す断面を作図するときに使用します。曲線の反対側の端点を選ぶと位置や向きが変わります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　WCS X 角度	コマンド ID：DrawingPlane.WcsXAngle

ワールド座標 X 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　ワールド座標の X 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：WCS の X 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　WCS Y 角度	コマンド ID：DrawingPlane.WcsYAngle

ワールド座標 Y 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　ワールド座標の Y 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：WCS の Y 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　WCS Z 角度	コマンド ID：DrawingPlane.WcsZAngle

ワールド座標 Z 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　ワールド座標の Z 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：WCS の Z 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　UCS X 角度	コマンド ID：DrawingPlane.UcsXAngle

現在の作業平面 X 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　現在の作業平面の X 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：UCS の X 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　UCS Y 角度	コマンド ID：DrawingPlane.UcsYAngle

現在の作業平面 Y 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　現在の作業平面の Y 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：UCS の Y 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：モデリング
 コマンド名：　UCS Z 角度	コマンド ID：DrawingPlane.UcsZAngle

現在の作業平面 Z 軸を基準軸として、指定角度の作業平面を作成します。

操作手順：

1　現在の作業平面の Z 軸を基準にする角度コマンドを実行します。

2　角度を入力します。

3　回転後の作業平面と UCS の向きを確認して確定します。

プロパティエクスプローラー：

■ 角度：UCS の Z 軸を基準にした回転角度です。

操作のポイント：

WCS は図面全体の固定座標、UCS は現在の作業平面の座標です。作業平面を傾けた後は、同じ軸名でも基準方向が異なります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　移動	コマンド ID：JDesign.Modify.Move

選択した要素を移動します。

操作手順：

1　移動する要素を選択してコマンドを実行します。

2　移動元の基準となる始点を指示します。

3　移動先の終点を指示すると、始点が終点へ重なるように移動します。

プロパティエクスプローラー：

■ 始点・終点：移動元と移動先の基準点です。

■ 距離：始点から終点方向への移動量を数値で指定できます。

操作のポイント：

元の位置にも残したい場合は「複写」を使用します。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　複写	コマンド ID：JDesign.Modify.Copy

選択した要素を連続複写します。

操作手順：

1　複写する要素を選択してコマンドを実行します。

2　複写元の起点を指示します。

3　複写先起点を指示します。続けて別の位置を指定すると連続複写できます。

4　必要な複写を終えたら Esc キーで終了します。

プロパティエクスプローラー：

■ 起点：複写元の基準点です。

■ 複写先起点：コピーを置く基準点です。

■ 距離：複写先方向への距離を入力できます。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　2D ストレッチ	コマンド ID：JDesign.Modify.Stretch

選択した要素の端点や頂点をストレッチします。

操作手順：

1　対象となる 2D 要素を選択してコマンドを実行します。

2　範囲の対角 2 点を指示し、動かす端点や頂点を囲みます。

3　基点と移動先を指示します。

4　動く点と残る点をプレビューで確認して確定します。

プロパティエクスプローラー：

■ 範囲 1 点目・範囲 2 点目：移動対象の点を囲む矩形を画面上で指定します。

■ 基点・移動先：ストレッチする方向と距離を指定します。

操作のポイント：

要素全体の移動とは異なります。残したい側の端点を範囲へ入れないようにしてください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　回転	コマンド ID：Object.Rotate

選択オブジェクトを軸を中心に回転します

操作手順：

1　回転する要素を選択してコマンドを実行します。

2　回転の基準点を指定します。任意軸を使う場合は軸となる線分を選択します。

3　回転角度を入力し、向きと複写の有無を確認して確定します。

プロパティエクスプローラー：

■ 基準点・軸：回転中心を決めます。

■ 角度：回転量を指定します。始点・終点による方向指定も使用できます。

■ 元平面・先平面：平面を基準に姿勢を合わせる場合に指定します。

■ 複写：元要素を残すかどうかを設定します。

操作のポイント：

これは既存の要素を回すコマンドです。断面から回転体を作るモデリングの「回転」とは異なります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　鏡象	コマンド ID：Object.Reflect

選択オブジェクトを指定した軸または平面で反転します

操作手順：

1　反転させる要素を選択します。

2　反転の基準となる点、線分、または平面を指定します。

3　元要素を残すかどうかを設定し、反転後の向きを確認して確定します。

プロパティエクスプローラー：

■ 点・線分・平面：目的に合う反転基準を指定します。

■ 複写：元要素を残して鏡像を作る場合に選択します。

操作のポイント：

左右対称の部品は、対称軸や対称面をスナップで正確に指定します。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　尺度変更	コマンド ID：Object.Scale

選択オブジェクトを拡大縮小します

操作手順：

1　拡大・縮小する要素を選択します。

2　動かさない固定点を指定します。

3　倍率を入力し、元要素を残すか確認して確定します。

プロパティエクスプローラー：

■ 固定点：拡大・縮小の中心です。

■ 倍率：2 なら 2 倍、0.5 なら半分になります。

■ 始点・終点：基準長さから倍率を決める場合に使用します。

■ 変形・X/Y/Z 倍率：方向別に倍率を変える場合の設定です。

■ 複写：元要素を残すかを選びます。

操作のポイント：

印刷尺度やレイアウトの尺度変更とは異なり、図形そのものの寸法が変わります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　削除	コマンド ID：Object.Delete

選択オブジェクトを削除します

操作手順：

1　削除する要素を選択します。

2　選択色になった対象を確認し、コマンドを実行します。

操作のポイント：

表示だけを隠す操作ではありません。誤って削除した直後は「元に戻す」で確認してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　分解	コマンド ID：SelectedObjects.Decompose

選択した要素のうち、分解可能なブロックやポリラインなどを分解します

操作手順：

1　クリックまたは囲み選択で、分解する要素をまとめて選択します。

2　コマンドを実行します。

3　個々の構成要素として選択できることを確認します。

操作のポイント：

一階層だけ分解します。ダイナミックブロックは現在の形状に分かれ、内部の入れ子のブロック参照は残ります。編集用の原本は複写して残してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　オフセット	コマンド ID：Aequidist

等距離曲線を作成します

操作手順：

1　等距離曲線を作る基準曲線を選択します。

2　距離を入力し、内側・外側のどちらへ作るか確認します。

3　角部の扱いを設定して確定します。

プロパティエクスプローラー：

■ 基準曲線：オフセットする元の曲線です。

■ 距離：基準曲線からの符号付き距離です。

■ 鋭角にする最小角度：0°はすべて鋭角、180°は鋭角なしの設定です。

操作のポイント：

細い部分に対して大きすぎるオフセットを指定すると、自己交差などで形状を作れない場合があります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　トリム	コマンド ID：Trim

交点で要素をトリムします

操作手順：

1　必要に応じて「トリム先」で切断の基準要素を選択します。

2　「トリム」で削除したい区間の側をクリックします。

3　交点まで短くなった形状を確認します。

プロパティエクスプローラー：

■ トリム先：切りそろえる基準です。省略する操作では周囲の交点を使用します。

■ トリム：切り取る要素と区間を指定します。

操作のポイント：

残す側ではなく、取り除きたい部分を指示してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　サーフェストリム	コマンド ID：Constr.Face.SurfaceTrim

交差するサーフェスで対象サーフェスをトリムします

操作手順：

1　トリムする対象サーフェスを選択します。

2　対象面と交差する切断サーフェスを選択します。

3　交線で分かれた領域から、削除する側を選択して実行します。

プロパティエクスプローラー：

■ 対象サーフェス：加工される面です。

■ 切断サーフェス：境界を作る面です。

■ 削除する領域：取り除く側を指定します。

操作のポイント：

サーフェス同士が実際に交差する必要があります。画面上で重なって見えるだけでは切断できません。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　分割	コマンド ID：Trim.Split

交点で要素を分割します

操作手順：

1　分割する曲線の、交点に近い位置を指示します。

2　分割位置を確認して実行します。

3　分かれた要素を個別に選択できることを確認します。

プロパティエクスプローラー：

■ 分割：対象となる要素と位置を指定します。

操作のポイント：

「トリム」と異なり、分割した両側を残して別の要素にします。

タブメニュー：　ホーム	グループ：修正
コマンド名：　結合	コマンド ID：Join

同一レイヤ・同一色・同一線種で端点が一致する選択曲線を連鎖ごとに結合します

操作手順：

1　つなぎたい線分・円弧などをまとめて選択します。

2　コマンドを実行します。

3　端点がつながった連鎖ごとに結合されたことを確認します。

操作のポイント：

同一レイヤ・同一色・同一線種で、端点が一致する曲線が対象です。隙間や属性の違いがある場合は先にそろえてください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　重なり要素を除去	コマンド ID：RemoveOverlappingCurves

同一レイヤ・同一色・同一線種・同一線幅で重複または重なっている選択曲線を 1 本に統合します

操作手順：

1　重複を整理する曲線を選択します。

2　コマンドを実行します。

3　重なっていた区間が整理されたことを確認します。

操作のポイント：

同一レイヤ・同一色・同一線種・同一線幅の曲線が対象です。異なる属性の線を無条件に削除する操作ではありません。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　延長	コマンド ID：Expand

要素を延長します

操作手順：

1　延長したい要素を、伸ばす端に近い位置で選択します。

2　延長先の要素を指定します。必要なら追加距離を設定します。

3　延長結果を確認して確定します。

プロパティエクスプローラー：

■ 延長する要素：端を伸ばす曲線です。

■ 延長先：到達先となる要素です。

■ 距離：さらに伸ばす量です。負の値は短くする方向になります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　包絡	コマンド ID：JDesign.Modify.Envelope

矩形範囲内の直線を包絡処理します。

操作手順：

1　包絡処理する範囲の 1 点目を左クリックします。

2　対角の 2 点目を左クリックして、範囲内の直線を処理します。

3　結果を確認します。

マウス操作：

■ 操作：2 点目を右クリックすると範囲内削除、Shift＋左クリックでは中間部分の消去になります。

操作のポイント：

2 点目のボタン操作で処理内容が変わります。通常の包絡は左クリックで指定してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　クリップ円形	コマンド ID：JDesign.ClipCopy.Circle

円形境界の内側を切り取り、指定位置へ複写します

操作手順：

1　必要に応じて複写対象の 2D 要素を選択します。

2　円形範囲の中心と半径を指定します。

3　複写先を指示し、円内の図形を配置します。

指定する項目：

■ 中心・半径：切り出す円形範囲です。

■ 複写先：切り出した内容を置く位置です。

操作のポイント：

元図形を残したまま複写します。文字や寸法などは、境界内に完全に含まれるものが対象です。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　クリップ矩形	コマンド ID：JDesign.ClipCopy.Rectangle

矩形境界の内側を切り取り、指定位置へ複写します

操作手順：

1　必要に応じて複写対象の 2D 要素を選択します。

2　矩形範囲の対角 2 点を指示します。

3　複写先を指示して配置します。

指定する項目：

■ 対角 2 点：切り出す矩形範囲です。

■ 複写先：切り出した内容を置く位置です。

操作のポイント：

境界をまたぐ曲線は範囲内を切り出します。文字や寸法は完全に含まれるものだけを複写します。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　フィレット	コマンド ID：Round.Off

共通頂点で要素を丸めます

操作手順：

1　丸める角の近くで要素を選択します。必要なら第 2 要素を指定します。

2　半径を入力し、元要素をトリムするか設定します。

3　接続する円弧と残る線の範囲を確認して確定します。

プロパティエクスプローラー：

■ 位置・第 2 要素：丸める 2 つの要素を指定します。

■ 半径：角丸の大きさです。

■ トリム：元の要素を円弧の接点まで切りそろえるかを指定します。

操作のポイント：

2D 曲線の角丸です。ソリッドの稜線には「3D フィレット」を使用します。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　全角フィレット	コマンド ID：RoundMultiple

すべての共通頂点で要素を丸めます

操作手順：

1　角をまとめて丸めるポリラインやパスを選択します。

2　半径を入力します。

3　各角の丸め結果を確認して確定します。

プロパティエクスプローラー：

■ 位置：対象のパスや連続した要素を指定します。

■ 半径：角に付ける円弧の半径です。

操作のポイント：

短い辺や狭い角では、同じ半径を付けられない場合があります。すべての角を確認してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　面取り	コマンド ID：Cut.Off

共通頂点で要素を面取りします

操作手順：

1　面取りする角の要素を選択します。必要なら第 2 要素を指定します。

2　モード、長さ、角度を設定します。

3　元要素のトリム設定と面取り線を確認して確定します。

プロパティエクスプローラー：

■ 位置・第 2 要素：面取りする要素です。

■ モード：入力する長さの扱いを選択します。

■ 長さ・角度：面取りの大きさと傾きです。

■ トリム：元要素を切りそろえるかを設定します。

操作のポイント：

2D 図形を加工するコマンドです。ソリッドのエッジには 3D 面取りを使います。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　全角面取り	コマンド ID：CutOffMultiple

すべての共通頂点で要素を面取りします

操作手順：

1　角をまとめて面取りするパスなどを選択します。

2　モード、長さ、角度を設定します。

3　全体の形状を確認して確定します。

プロパティエクスプローラー：

■ 位置：面取りする対象です。

■ モード・長さ・角度：各角へ適用する面取り条件です。

操作のポイント：

一部の角だけを加工するときは、通常の面取りコマンドで個別に指定してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　3D フィレット	コマンド ID：Fillet

3D フィレットコマンドを実行します。

操作手順：

1　フィレットを付けたいエッジを選択します。複数の稜線を指定することもできます。

2　半径を入力します。面やソリッドからまとめて選ぶ場合は対象範囲を確認します。

3　選択した稜線と半径を確認して実行します。

プロパティエクスプローラー：

■ エッジ：個別にフィレットを付ける稜線です。

■ 面のエッジ：選択面に接する稜線を対象にします。

■ ソリッドのエッジ：シェルやソリッドの稜線を対象にします。

■ 半径：フィレットの大きさを指定します。

操作のポイント：

面全体から選ぶと、意図しない稜線まで含む場合があります。局所的に付けるときはエッジを個別に選びます。半径が周囲の厚みや隙間に対して大きすぎる場合は作成できません。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　3D 面取り	コマンド ID：Chamfer

ソリッドのエッジに面取りを作成します

操作手順：

1　面取りの基準面と対象エッジを指定します。

2　等辺面取りまたは不等辺面取りを選択します。

3　距離を入力し、基準面側と反対側を確認して実行します。

プロパティエクスプローラー：

■ モード：等辺・不等辺を切り替えます。

■ 基準面・エッジ：面取りする位置を指定します。

■ 距離 1：基準面側の距離です。

■ 距離 2：反対側の距離です。

操作のポイント：

不等辺面取りでは基準面の選び方で距離 1 と距離 2 の側が変わります。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　矩形配列複写	コマンド ID：Copy.Matrix

選択オブジェクトの複数コピーをグリッド状に配置します

操作手順：

1　配列複写する要素を選択します。

2　上下左右のコピー数と、水平・垂直距離を入力します。

3　グリッドの角度を設定し、配置全体を確認して実行します。

プロパティエクスプローラー：

■ 右・左・上・下方向コピー数：元要素から各方向に作る数です。

■ 水平距離・垂直距離：隣り合うコピーの間隔です。

■ 角度：配列全体の向きを指定します。

操作のポイント：

個数を大きくすると作成要素が増えます。まず少ない数で方向と間隔を確認してください。

タブメニュー：　ホーム	グループ：修正
 コマンド名：　円形配列複写	コマンド ID：Copy.Circular

選択オブジェクトの複数コピーを円周上に配置します

操作手順：

1　配列複写する要素を選択します。

2　配列の中心を指示し、数を入力します。

3　全円配置または角度ステップを選び、必要な角度を指定します。

4　コピーの向きを固定するか回転させるか確認して実行します。

プロパティエクスプローラー：

■ 中心：配列の回転中心です。

■ 数・配置：コピー数と全円／角度ステップを指定します。

■ 角度：角度ステップで隣り合うコピーの間隔を設定します。

■ 向きのモード：固定／回転を選択します。

操作のポイント：

穴の配置では中心を正確に指定し、最後のコピーが元要素に重ならないことを確認します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　和	コマンド ID：Constr.Solid.Fuse

重なっている選択ソリッドを和演算で結合します

操作手順：

1　結合するソリッドを複数選択します。

2　コマンドを実行します。

3　重なり部分が整理され、結合されたことを確認します。

操作のポイント：

線やサーフェスをまとめる操作ではありません。対象が体積を持つソリッドで、互いに重なる位置にあるか確認します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　差	コマンド ID：Constr.Solid.Difference

1 つ目の選択ソリッドから 2 つ目の選択ソリッドを差し引きます

操作手順：

1　削られる側のソリッドを最初に選択します。

2　切削に使うソリッドを次に選択します。

3　差演算を実行し、重なった部分が取り除かれたことを確認します。

操作のポイント：

選択順に意味があります。対象と工具を逆にすると結果も逆になります。2 つが交差していない場合は目的の穴は開きません。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　交差	コマンド ID：Constr.Solid.Intersect

選択ソリッドを交差部分のソリッドに置き換えます

操作手順：

1　共通部分を残すソリッドを選択します。

2　交差演算を実行します。

3　両方に含まれる部分だけが残っていることを確認します。

操作のポイント：

和演算とは逆に、重ならない部分は残しません。共通する体積がある形状で使用します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　シェル	コマンド ID：Constr.Solid.Shell

指定した解放面を取り除き、一定の厚みでソリッドをくり抜きます

操作手順：

1　開口として取り除く面を選択します。必要なら複数面を指定します。

2　残す壁の厚みを入力します。

3　開口と内側の形状を確認して実行します。

プロパティエクスプローラー：

■ 解放面：シェル化で取り除く面です。

■ 厚み：残す肉厚です。

操作のポイント：

狭い部分や曲率の小さい部分では、指定した厚みで内側の面を作れない場合があります。

操作場所： ホーム	グループ：モデル修正
 穴形状 基本形状と入口	操作画面：穴形状フォーム

概要

面上の指定位置に穴を開けます。「ホーム」→「モデル修正」→「シェル」の隣の大ボタン「穴形状」から起動します。フォームを開いたまま CAD 画面を操作できます。

操作手順

1　「穴の形状」タブで種類、穴径、終端、穴底、入口を設定します。

2　「配置」タブで配置方法を選び、CAD 画面上で位置を指定します。

3　緑色の位置円と配置個数を確認します。

4　「適用して続ける」で作成して次の配置へ進みます。「作成して閉じる」は作成後にフォームを閉じます。

穴の種類

丸穴：穴径を直接指定します。ボルト通し穴：呼びからキリ穴径と座ぐり寸法を設定できます。タップ下穴（ねじ情報）：下穴にねじ情報・簡略表示を付けます。リーマ穴（公差情報）：穴径と公差の情報を指定します。

終端と穴底

「貫通（選んだソリッド）」または「止まり」を選びます。止まりの深さはドリル先端を除いた円筒部の長さです。穴底はドリル先端または平底とし、ドリル先端では先端角度を入力します。

入口の設定

なし・座ぐり・皿もみ・スポットフェースから選びます。座ぐりはキリ穴径 d′、座ぐり径 D、深さ H2、皿もみは入口径と皿角度を設定します。ボルト呼びは M1.6〜M30 の登録値から選択できます。

入力例

M6 の登録値はキリ穴径 6.6mm、座ぐり径 11mm、座ぐり深さ 6.5mm です。「任意寸法」では各値を直接入力できます。採用するボルトと加工条件に合わせて確認してください。

操作場所： ホーム	グループ：モデル修正
 **穴形状 連続クリックと配列配置**	操作画面：穴形状フォーム 配置

連続クリック

1　「配置方法」で「連続クリック」を選びます。

2　CAD 画面上で穴の中心を順にクリックします。クリックした面の位置と局所法線から、各穴の位置・方向が決まります。

3　異なる面にも続けて配置できます。後のクリックで、すでに配置した穴の方向は変わりません。

円形連続配置

1　「円形連続配置」を選び、個数・PCD・開始角度・配置角度を設定します。

2　基準にする平面上で、配列の中心をクリックします。

3　360 度は一周の等間隔配置で、終点に穴を重複させません。負の配置角度では逆回りになります。

矩形連続配置

1　「矩形連続配置」を選び、X 方向・Y 方向の個数と間隔、配列の回転角度を設定します。

2　平面上で始点をクリックします。個数には始点を含みます。一方向を 1 個にすると一列の配置になります。

向きと対象

配列の 0 度は基準面に投影したワールド X 方向です。ワールド X に垂直な面ではワールド Y 方向を使います。円形・矩形配置は基準平面上の配列です。

配置の確認と取消

緑色の円と一覧で位置を確認します。「最後を取り消す」で直前の配置を戻し、「配置をクリア」で未作成の配置を消します。個数は 6 個固定ではありません。

境界にまたがる穴

工具形状を面の輪郭で切り取らないため、縁にまたがる穴や半穴も作れます。差し引く対象は指定面が属するソリッドです。非表示・ロック中のソリッドは対象外です。

操作場所： ホーム	グループ：モデル修正
 **穴形状 ねじ情報と簡略表示**	操作画面：穴形状フォーム ねじ

概要

下穴にねじの呼び・ピッチなどを保存し、モデルとレイアウトに簡略表示を付けます。らせん状のねじ山を持つ立体形状は作成しません。

操作手順

1　「タップ下穴（ねじ情報)」を選び、「ねじ」タブを開きます。

2　「ねじ情報と簡略表示を有効にする」をオンにし、ねじの呼びとピッチ P を選びます。

3　ねじ深さ、入口からの開始位置、ねじの向き、公差等級を設定します。

4　モデルとレイアウトの表示を必要に応じてオンにし、穴を配置して作成します。

寸法データ

呼びとピッチに対応する基本寸法を ANASIZE.dat から読み込みます。同じ呼びに複数ピッチがある場合は先頭の登録値が初期値です。「基本の内径 D1 を下穴径に設定」は基本寸法を転記する操作であり、加工用ドリル径の推奨値ではありません。

モデルの表示

「モデルにねじ模様を表示」で穴内面に沿った簡略線を表示します。座ぐり・皿もみ部分は表示範囲から除きます。公差等級とねじの向きはねじ情報として扱います。

レイアウトの表示

「レイアウトにねじ製図線を表示」で、軸方向の図には内外径の二重円による表現、側面にはねじ範囲を示す線を追加します。隠れる部分は隠れ線で表します。不完全ねじ部の表示は「ねじ深さ」タブで設定します。

操作のポイント

「円筒部の終端までねじを表示」をオンにすると穴の終端まで表示します。ねじ深さを個別指定する場合はこの設定をオフにし、下穴深さとの関係も確認してください。

操作場所： ホーム	グループ：モデル修正
 穴形状 ねじ深さの自動計算	操作画面：穴形状フォーム ねじ深さ

概要

ねじ深さと下穴深さを目安値から設定し、必要な項目だけ手入力で変更できます。D はねじ呼びの直径、P はピッチです。自動値は加工の目安であり、強度や工具条件を判定するものではありません。

操作手順

1　「穴の形状」で止まり穴を選び、「ねじ」で呼びとピッチを設定します。

2　終端までのねじ表示をオフにし、「ねじ深さ」タブでねじ深さの基準を選びます。

3　ねじ深さ、不完全ねじ部、底の余裕、下穴の円筒部と結果表示を確認します。

4　変更したい数値を直接入力し、「適用して続ける」または編集時の「変更を適用」で確定します。

自動値の基準

ねじ深さは 1D・1.5D・2D から選択でき、初期値は 2D です。不完全ねじ部は 2.5P、底の余裕は 1P を初期値とします。工具先端の追加長さは工具に合わせて指定します。

下穴の深さ

自動計算する円筒部の深さは、ねじの開始位置＋ねじ深さ＋不完全ねじ部＋底の余裕＋工具先端の追加長さです。開始位置は「入口からの開始」と入口加工の終わりのうち、深い方を基準にします。ドリルの円すい先端は円筒部深さの外側に付加されます。

計算例

入口加工なし・開始 0mm・M6×1・2D・工具先端の追加 0mm では、ねじ深さ 12mm、不完全ねじ部 2.5mm、底の余裕 1mm、下穴円筒部 15.5mm が目安になります。

手動変更と自動復帰

数値を直接入力すると、その項目だけ手動になります。「自動：2D」「自動：2.5P」「自動：1P」「自動計算」で対応する項目を自動に戻せます。手動の下穴が短い場合は、ねじ範囲が収まる値へ見直してください。

フランジの中央穴を中心に円形配置する

1. 穴コマンドで円形配置を選び、個数・配列径などを設定します。

2. 穴を加工する平面をクリックします。この段階では位置は未確定です。

3. 中心スナップを使い、中央穴の中心を配列中心に指定します。加工面は先に選んだ平面のままです。

4. 配置と加工方向を確認して確定します。矩形配置も、平面の選択後に基準位置を指定します。

中心を指定し直すクリックは中心位置だけを変えます。平面から選び直す場合は［配置をクリア］、一つ前へ戻る場合は［最後を取り消す］を使います。

穴を再編集する

穴コマンドで作った穴の内面・段面・穴底をダブルクリックすると、同じ操作で作った穴の編集画面が開きます。入力後に［変更を適用］または［変更して閉じる］で確定します。穴に影響しない別加工は保持します。

入口側・裏側の平面を穴軸方向へ移動した場合も、対応する形状では変更後の厚みを保って編集できます。穴口にフィレットや切削が重なるなど、再構成できない場合は元形状を保持します。履歴のない STEP の穴を穴コマンドの履歴へ変換する機能ではありません。

操作場所： ホーム	グループ：モデル修正
 **穴形状 ダブルクリック編集**	操作画面：穴形状フォーム 編集

概要

穴形状コマンドで作成した穴をダブルクリックし、寸法やねじ情報を変更します。同じ 1 回の操作でまとめて作成した穴は、同じ編集対象として扱います。

操作手順

1　通常の選択状態で、穴の内面・座ぐりの段面・穴底をダブルクリックします。

2　編集フォームが開いたら、ハイライトされた穴と編集対象の個数を確認します。

3　穴径、入口形状、深さ、ねじパラメーターなどを変更します。

4　「変更を適用」で変更して編集を続けます。「変更して閉じる」は適用後にフォームを閉じます。

対象と配置

対象の面と位置円が固定表示され、マウスを動かしても編集対象は移りません。編集時は配置位置と穴の方向を固定して、形状のパラメーターを変更します。

保存と移動

編集情報は JDC に保存されます。立体全体のドラッグ移動・回転後や、コピーした立体の穴も編集できます。コピー側を変更しても元の立体は変更されません。

編集できない場合

穴の後に面移動やフィレットなど別の形状加工を行うと、記録した穴形状と一致せず、再編集できない場合があります。通常の押し出し差演算など、穴形状コマンド以外で作った穴も対象外です。

適用と取消

「閉じる」は未適用の入力を破棄します。処理中は「計算を中止」で中断できます。処理に失敗した場合は結果を適用せず、入力モデルを保持します。確定した変更は「元に戻す」で戻せます。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　断面線	コマンド ID：PlaneIntersection

ソリッド、シェル、面を平面で交差させます

操作手順：

1　断面線を求めるソリッド、シェル、面を選択します。

2　交差に使う平面を指定します。

3　交線を確認して確定します。

プロパティエクスプローラー：

■ 平面：断面を取り出す位置と向きを指定します。

操作のポイント：

交線を作成する操作です。元の立体を 2 つに分ける場合は「ソリッド－面切断」を使用します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　ソリッド-面切断	コマンド ID：Constr.Solid.SplitByFace

選択したソリッドをフェースまたは開いたシェルで切断し、両側のソリッドを残します

操作手順：

1　切断するソリッドを指定します。

2　切断フェース、または開いたシェル上のフェースを指定します。

3　切断位置を確認して実行します。

プロパティエクスプローラー：

■ ソリッド：切断される立体です。

■ 切断フェース：切断の境界を作る面です。

操作のポイント：

切断後は両側のソリッドを残します。片側を取り除く場合は、結果を確認して不要な側を削除します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　ソリッド-干渉チェック	コマンド ID：Constr.Solid.InterferenceCheck

ソリッド-干渉チェックコマンドを実行します。

操作手順：

1　干渉を確認するソリッドを複数選択します。

2　干渉チェックを実行します。

3　表示された結果から、部品が重なっている箇所を確認します。

操作のポイント：

接している状態と、体積が重なっている状態は異なります。部品配置を変更した場合は再度チェックします。

切断と干渉チェックの結果を確認する

ソリッド切断では対象のソリッドと切断面を指定します。面が対象を横切っていることを確認し、切断後は各部品を選択して意図した位置で分かれているか確認します。

干渉チェックでは対象ソリッドを選びます。元のソリッドは元の画層に残り、干渉部分の結果を「干渉チェック」画層へ作成します。再実行時も元の部品と結果の図形を区別してください。

DWG から読み込んだ塗りつぶし記号などを選択するときも、画層のロックと選択フィルターを確認します。元のソリッドや編集情報がすでに失われている場合、ソフトを更新しただけでは復元されません。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　フィレット	コマンド ID：Fillet

フィレットコマンドを実行します。

操作手順：

1　フィレットを付けたいエッジを選択します。複数の稜線を指定することもできます。

2　半径を入力します。面やソリッドからまとめて選ぶ場合は対象範囲を確認します。

3　選択した稜線と半径を確認して実行します。

プロパティエクスプローラー：

■ エッジ：個別にフィレットを付ける稜線です。

■ 面のエッジ：選択面に接する稜線を対象にします。

■ ソリッドのエッジ：シェルやソリッドの稜線を対象にします。

■ 半径：フィレットの大きさを指定します。

操作のポイント：

面全体から選ぶと、意図しない稜線まで含む場合があります。局所的に付けるときはエッジを個別に選びます。半径が周囲の厚みや隙間に対して大きすぎる場合は作成できません。

切断・面取り・フィレットが成立しない場合

切断面がソリッドを横切っているか、穴や突起との境界、面の交差、極端に細い残り厚みを確認します。面取り幅・フィレット半径は周囲の形状に制限されます。小さい値なら必ず成立するわけではありません。

切断・面取り・フィレット・突起編集の結果は確定後に確認します。過去の演算で壊れた形状や消失した突起を、図面を開くだけで自動復元するものではありません。演算前の保存ファイルを残し、失敗箇所を確認してください。

複数エッジのフィレットと接続を確認する

管とフランジの境界では、対応する形状に限り、指定半径の丸みを側面の境界で切り止めて作成できます。通常のフィレットで境界エッジと半径を指定します。端部がフランジの外へ突出しないこと、穴や隣接部品が保たれることを断面でも確認してください。

丸みを変更するときは［フィレットサイズ変更］でフィレット面を選びます。削除は［フィレット削除］を使用します。対応する管の分岐・円形フランジ・段付き管の丸みも対象です。復元可能な支持面と形状条件が必要で、任意の半径・全ての読込形状で成立するわけではありません。確定後は Undo／Redo、保存と再読込で結果を確認できます。

フィレットを入れる境界エッジを選択し、半径を指定してチェックで確定します。同じ半径の辺はまとめて選択できます。面そのものではなく、丸めたい二つの面の境界を選んでいることを確認してください。

加工後は、選択した辺に丸みが付いていること、終端と隣接する丸みが滑らかにつながること、穴やリブが意図した形状を保つことを確認します。半径は周囲の肉厚、面間の距離、丸み同士の干渉によって制限されます。小さい半径なら必ず成立するわけではありません。

複数の丸みを削除するときは、戻したいフィレット面を指定して結果を確認します。対応する形状では支持面を延長して元の角へ戻します。元の面の復元が成立しない場合は無理に続けず、操作前の図面や Undo を利用してください。

円形フランジでは円周の内外と上下を確認して選びます。中央穴の上端へ等辺面取りを入れる場合は穴の上端の内側エッジを指定し、距離を入力します。内側と外側の縁を取り違えないよう、断面表示も利用してください。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面取り	コマンド ID：Chamfer

ソリッドのエッジに面取りを作成します

操作手順：

1　面取りの基準面と対象エッジを指定します。

2　等辺面取りまたは不等辺面取りを選択します。

3　距離を入力し、基準面側と反対側を確認して実行します。

プロパティエクスプローラー：

■ モード：等辺・不等辺を切り替えます。

■ 基準面・エッジ：面取りする位置を指定します。

■ 距離 1：基準面側の距離です。

■ 距離 2：反対側の距離です。

操作のポイント：

不等辺面取りでは基準面の選び方で距離 1 と距離 2 の側が変わります。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　フィレット削除	コマンド ID：Fillet.Remove

フィレット面を削除して角形状を戻します

操作手順：

1　取り除くフィレットの面を選択します。

2　周囲のどの面を延ばして戻すかを確認します。

3　実行して角の形状が復元されたことを確認します。

プロパティエクスプローラー：

■ 除去するフィレット面：丸みを構成する面を指定します。

操作のポイント：

エッジではなくフィレット面を選択します。後の加工で周囲の面が変わっていると、復元できない場合があります。

操作場所： ホーム	グループ：モデル修正
 **フィレットサイズ変更**	コマンド ID：Fillet.ChangeRadius

概要

作成済みの 3D フィレットを選び、半径を変更します。新たに角を丸めるフィレット作成とは別のコマンドです。

操作手順

1　「ホーム」→「モデル修正」→「フィレット」の▼→「フィレットサイズ変更」を選びます。

2　変更したいフィレット面をクリックします。稜線ではなく、丸みの面を選びます。

3　「変更後の半径」に新しい R 寸法を入力します。最初に選択した箇所の現在半径が初期値になります。

4　選択面のハイライトと数値を確認してチェックマークで確定します。

複数箇所の選択

同じ半径で滑らかにつながる面はまとめて選択されます。離れたフィレットや別のソリッドも追加できます。選択したグループを再度クリックすると解除できます。

処理中の操作

入力中は対象のハイライトを表示し、確定時に形状を計算します。計算中は進行表示から、または Esc キーで中止できます。まとめて変更した結果は 1 回の「元に戻す」で戻せます。

操作のポイント

保存・再読込後の形状も認識対象です。ただし、すべての自由曲面や干渉形状を変更できるわけではありません。面を選択できない場合や変更が失敗する場合は、対象と半径を確認してください。失敗・中止時は元のモデルを保持します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面の除去	コマンド ID：Face.Remove

開口部の面を除去し、可能な場合は周囲の面を復元します。

操作手順：

1　取り除く開口部などの面を選択します。

2　対象面を確認して実行します。

3　周囲の面が復元され、不要な開口が残っていないか確認します。

プロパティエクスプローラー：

■ 除去する面：形状から取り除く面です。

操作のポイント：

単に面を非表示にする操作ではありません。周囲の面で補えない形状は処理できない場合があります。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面の移動	コマンド ID：Face.Move

選択した面を移動してソリッド形状を変更します。

操作手順：

1　移動する穴や突起などの面を選択します。

2　移動元の基点を指示します。

3　移動先を指示し、周囲の形状とのつながりを確認して実行します。

プロパティエクスプローラー：

■ 移動する面：位置を変更する部分です。

■ 基点・移動先：移動の方向と量を指定します。

操作のポイント：

モデル全体の移動とは異なり、選んだ部分の位置を変更します。関連する面を取りこぼさないようにします。

複数ソリッドの面をまとめて編集

［面の移動］［面のオフセット］［面のテーパー］は、異なるソリッドの面を同時に指定できます。ソリッド同士を「和」で結合しておく必要はありません。

高さをそろえる操作例

1. 対象が分かる等角図などへ切り替え、［面の移動］を実行します。

2. 複数の柱の上面をクリックまたは矩形選択で選びます。側面まで選んでいないか確認します。

3. 柱上端の角を基点、そろえる梁上端の角を移動先に指定します。端点スナップの取得位置を確認します。

4. 確定後に各柱の上面とソリッドの閉じた形状を確認します。1 回の［元に戻す］でまとめて戻せます。

操作	基準と動き	選択時に確認する点
面の移動	共通の基点→移動先の方向	動かす面と固定する面の区別
面のオフセット	各面の外向き法線と距離	上面と下面では同じ正値でも逆方向
面のテーパー	共通の固定平面と角度	固定側の面・境界、形状の成立範囲

確定時の動作

入力中は選択面などを表示し、形状の計算は確定時に行います。すべての対象で計算に成功してから一括反映します。一つでも成立しない場合は、元の形状を保持して入力を修正できます。

ダイナミックブロックとの違い

通常の面編集は今回の形状を変更します。繰り返し使う部品で「長さ」の入力と面移動を結び付けたい場合は、第 3 部の 3D 面動作として登録します。登録したパラメーターと動作は JDC へ保存してください。

突起と穴を保ちながら面を移動する

「面の移動」で、移動する突起と穴を構成する面をまとめて選び、移動先を指定して確定します。対応する平面の組み合わせでは、突起の厚さ・穴径を保って根元の板を延長し、静止側のフィレットを保持します。

根元の面だけを選んだ場合とは選択範囲が異なります。移動・回転・複写後は穴、側面、根元のつながりを確認します。既存の丸みへ食い込む移動や、自己交差を生じる操作は成立しない場合があります。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面の回転	コマンド ID：Face.Rotate

選択した穴または突起の面を同一面上で回転します。

操作手順：

1　回転する穴または突起の面を選択します。

2　回転中心を指示します。

3　角度を指定し、配置と周囲の形状を確認して実行します。

プロパティエクスプローラー：

■ 回転する面：回転させる穴や突起です。

■ 回転中心：面を回す中心です。

■ 角度：回転する量です。

操作のポイント：

同一面上の穴や突起の配置を回すために使います。モデル全体の姿勢変更は修正の「回転」を使用します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面の複写	コマンド ID：Face.Copy

選択した穴や突起を複写します。

操作手順：

1　複写する穴または突起の面を選択します。

2　複写元の基点と複写先を指示します。

3　別の面へ置く場合は複写先面を指定します。

4　新しい形状を確認して実行します。

プロパティエクスプローラー：

■ 複写する面：複製する穴や突起です。

■ 基点・複写先：元と先の基準位置です。

■ 複写先面：別の面へ複写する場合の到達先です。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面のオフセット	コマンド ID：Face.Offset

選択した複数の面を各面の外向き法線方向へオフセットします。

操作手順：

1　オフセットする面を選択します。複数面を指定できます。

2　オフセット距離を入力します。

3　外側・内側のどちらへ変化するか確認して実行します。

プロパティエクスプローラー：

■ オフセットする面：位置を変える面です。

■ オフセット距離：各面の外向き法線を基準とした移動量です。

操作のポイント：

面ごとの法線方向へ動きます。複数面を同じ方向へ平行移動する操作とは異なります。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　面のテーパー	コマンド ID：Face.Taper

選択した複数の平面に共通のテーパー角度を付けます。

操作手順：

1　テーパーを付ける面を選択します。

2　中立面との境界になる基準エッジを選択します。

3　角度を入力するかギズモで指定し、傾く側を確認して実行します。

プロパティエクスプローラー：

■ テーパーを付ける面：加工する平面や対応する円筒面です。

■ 基準境界エッジ：中立面側の基準となる直線または円弧です。

■ テーパー角度：正の値では中立面から離れる側が内向きに傾きます。

操作のポイント：

基準境界を取り違えると、固定される側も変わります。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　曲線を面へ投影	コマンド ID：Curve.ProjectToFace

閉じたスケッチ曲線を曲面へ投影し、投影稜線で面を分割します。

操作手順：

1　閉じたスケッチ曲線を選択します。複数曲線なら端点がつながって閉じるようにします。

2　投影先の面を選択します。

3　投影方向と交点の選択方法を設定し、実行します。

プロパティエクスプローラー：

■ 投影する曲線：面へ写す輪郭です。

■ 投影先の面：投影稜線で分割する面です。

■ 投影方向：投影に使うベクトルです。

■ 交点の選択：投影線と面が複数回交わる場合の採用方法です。

操作のポイント：

投影した線で面を分割します。この操作だけで穴が開くわけではありません。

曲線を面へ投影する

閉じた曲線、投影先の面、投影方向を指定し、右上のチェックで確定します。投影方向は曲線から対象面へ届く向きにします。交差しない方向では面を分割できません。

複雑なフィレット面を含むモデルでは、確定後の計算や再描画に時間がかかる場合があります。完了後に、投影された境界、分割結果、ソリッドの閉じた形状を確認してください。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　断面平面を作成	コマンド ID：SectionPlane.Create

モデルに保存される断面平面を作成します（SECTIONPLANE）

操作手順：

1　面、3 点、または現在の作業平面を使って断面平面を指定します。

2　表示幅と表示高さを確認します。

3　確定して、図面内に断面平面オブジェクトを作成します。

プロパティエクスプローラー：

■ 断面平面：切断位置と方向を指定します。

■ 表示幅・表示高さ：画面に出す平面の枠の大きさです。

操作のポイント：

表示幅・高さは見た目の枠であり、切断計算をその範囲だけに制限する値ではありません。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　ライブ断面の ON/OFF	コマンド ID：SectionPlane.Live

選択した断面平面によるライブ断面表示を切り替えます（LIVESECTION）

操作手順：

1　保存されている断面平面の枠または矢印を選択します。

2　ライブ断面を切り替え、モデルの内側を確認します。

3　元の表示に戻すときは、同じ断面平面で再度切り替えます。

プロパティエクスプローラー：

■ 断面平面を選択：ライブ表示に使用する断面平面です。

操作のポイント：

表示を切り取る機能です。ソリッドの形状そのものを切断・削除する操作ではありません。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　断面線を XY 平面に	コマンド ID：SectionPlane.Generate2D

正確なソリッド交線を XY 平面上の 2D ブロックとして作成します（SECTIONPLANETOBLOCK）

操作手順：

1　2D 化する断面平面を選択します。

2　XY 平面上で 2D ブロックを置く位置を指示します。

3　確定し、生成された断面線を確認します。

プロパティエクスプローラー：

■ 断面平面を選択：交線を求める基準です。

■ 2D 配置点：作成するブロックの基点です。

操作のポイント：

ライブ断面の画面画像ではなく、ソリッドと平面の交線を 2D ブロックとして作成します。

タブメニュー：　ホーム	グループ：モデル修正
 コマンド名：　断面線設定	コマンド ID：SectionPlane.Settings

ライブ表示、切断方向、表示範囲を設定します（SECTIONPLANESETTINGS）

操作手順：

1　設定する断面平面の枠または矢印を選択します。

2　ライブ断面、切断方向、表示幅・高さを設定します。

3　表示される側を確認して確定します。

プロパティエクスプローラー：

■ ライブ断面：表示クリップの有効・無効です。

■ 切断方向を反転：残して見せる側を切り替えます。

■ 表示幅・表示高さ：断面平面の枠の大きさです。

操作のポイント：

表示範囲の設定と、実際のソリッド切断は別の操作です。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　自動寸法	コマンド ID：Constr.Automatic.Dimension

入力の仕方や対象要素で寸法を自動的に決定します

操作手順：

1　線分・円・円弧を選択するか、寸法の 1 点目を指示します。

2　長さを 2 点で指定する場合は 2 点目、角度の場合は 2 本目の線分を指定します。

3　マウスを移動して寸法の種類と位置を確認し、配置します。

プロパティエクスプローラー：

■ 対象／1 点目：要素選択と点指定を使い分けます。

■ 2 本目の線分／位置・2 点目：選んだ対象に応じて次の入力を行います。

■ 寸法線位置：寸法を置く場所です。

■ 寸法補助線角度：通常は寸法線に対して 90° です。

操作のポイント：

意図した種類にならない場合は、水平・垂直・半径などの専用寸法コマンドを使用します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　文字入力	コマンド ID：Constr.Text.1Point

文字: 基準点を指定します

操作手順：

1　文字の内容とフォント、大きさを設定します。

2　文字の基準点を指示します。

3　配置方向と位置を確認して確定します。

プロパティエクスプローラー：

■ 内容：図面へ記入する文字列です。

■ 文字フォント・文字高さ：書体と大きさを指定します。

■ 基準点：文字を配置する位置です。

操作のポイント：

寸法として計測値を表示したい場合は、文字を手入力するのではなく寸法コマンドを使います。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　文字入力 2 点指示	コマンド ID：Constr.Text.2Points

文字: 始点と終点を指定します

操作手順：

1　文字内容と書式を設定します。

2　文字の基準点を指示します。

3　基準線の第 2 点を指示し、文字の向きと表示を確認して確定します。

プロパティエクスプローラー：

■ 基準点：文字の配置基準です。

■ 第 2 点：文字基準線の方向を決めます。

■ 内容・フォント：文字列と書体を指定します。

操作のポイント：

斜めの線に沿った注記など、2 点で文字の向きを合わせたい場合に使用します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　文字輪郭パス化	コマンド ID：SelectedObjects.TextOutlinePath

文字輪郭パス化コマンドを実行します。

操作手順：

1　輪郭に変換する文字を選択します。

2　コマンドを実行します。

3　文字の外周や穴がパスとして取り出されたことを確認します。

操作のポイント：

変換後の輪郭は文字列としての編集とは扱いが異なります。元の文字も必要な場合は、複写を残してから変換してください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　マルチテキスト	コマンド ID：Constr.Text.MText

文字領域内に複数行文字を作成します

操作手順：

1　文字領域の 1 点目と対角点を指示します。

2　マルチテキストの編集画面で複数行の文章を入力します。

3　文字の折り返しと領域の大きさを確認して配置を確定します。

プロパティエクスプローラー：

■ 1 点目・対角点：文章を配置する領域です。

■ 文字内容・書式：編集画面で設定します。

操作のポイント：

長い注記は単一行文字を重ねるより、マルチテキストでまとめると編集しやすくなります。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　水平	コマンド ID：Constr.Horizontal.Dimension

水平の長さ寸法を作図します

操作手順：

1　寸法を測る 1 点目と 2 点目を指示します。

2　水平の寸法になることを確認し、寸法線位置を指定します。

3　続けて寸法を入れる場合はモードを選び、次の点を指示します。

プロパティエクスプローラー：

■ 点列：寸法補助線の起点となる点です。

■ 寸法線位置：寸法を図形から離して配置する位置です。

■ モード：単純・連続・直列・並列を選択します。

■ 寸法補助線角度：通常は 90° です。

■ 並列寸法間隔：並列寸法を配置するときの段間隔です。

操作のポイント：

円の中心間を測る場合は、両方とも中心スナップで指定してください。レイアウトでは対象ビューと表示される寸法値を確認します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　垂直	コマンド ID：Constr.Vertical.Dimension

垂直の長さ寸法を作図します

操作手順：

1　寸法を測る 1 点目と 2 点目を指示します。

2　垂直の寸法になることを確認し、寸法線位置を指定します。

3　続けて寸法を入れる場合はモードを選び、次の点を指示します。

プロパティエクスプローラー：

■ 点列：寸法補助線の起点となる点です。

■ 寸法線位置：寸法を図形から離して配置する位置です。

■ モード：単純・連続・直列・並列を選択します。

■ 寸法補助線角度：通常は 90° です。

■ 並列寸法間隔：並列寸法を配置するときの段間隔です。

操作のポイント：

円の中心間を測る場合は、両方とも中心スナップで指定してください。レイアウトでは対象ビューと表示される寸法値を確認します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　長さ	コマンド ID：Constr.Sloping.Dimension

平行した長さ寸法を作図します

操作手順：

1　寸法を測る 1 点目と 2 点目を指示します。

2　指定方向の寸法になることを確認し、寸法線位置を指定します。

3　続けて寸法を入れる場合はモードを選び、次の点を指示します。

プロパティエクスプローラー：

■ 点列：寸法補助線の起点となる点です。

■ 寸法線位置：寸法を図形から離して配置する位置です。

■ モード：単純・連続・直列・並列を選択します。

■ 寸法補助線角度：通常は 90° です。

■ 並列寸法間隔：並列寸法を配置するときの段間隔です。

■ 方向：長さを測る寸法線の方向を指定します。

操作のポイント：

円の中心間を測る場合は、両方とも中心スナップで指定してください。レイアウトでは対象ビューと表示される寸法値を確認します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　要素間	コマンド ID：Constr.Distance.Dimension

2 つの要素間の距離寸法を作図します

操作手順：

1　距離を測る第 1 要素を選択します。

2　第 2 要素を選択します。

3　測定される位置と寸法線を確認し、寸法線位置を指定します。

プロパティエクスプローラー：

■ 第 1 要素・第 2 要素：距離を測る対象です。

■ 寸法線位置：寸法を配置する場所です。

■ 寸法補助線角度：寸法線に対する補助線の傾きです。

操作のポイント：

要素の選択位置によって求める距離の候補が変わる場合があります。意図した箇所を測っているか確認します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　半径	コマンド ID：Constr.Radius.Dimension

半径寸法を作図します

操作手順：

1　半径を記入する円または円弧を選択します。

2　マウスを移動して半径寸法の文字位置を決めます。

3　R の付いた寸法値と引出し方向を確認して配置します。

プロパティエクスプローラー：

■ 円 or 円弧：半径を測る対象です。中心点ではなく円周・円弧を選択します。

■ 寸法線位置：文字と引出し線の配置を指定します。

操作のポイント：

中心が画面外にある大きな円弧でも、表示されている円弧から対象を選びます。寸法値は文字を手入力して合わせず、対象と尺度を確認してください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　直径	コマンド ID：Constr.Diameter.Dimension

直径寸法を作図します

操作手順：

1　直径を記入する円または円弧を選択します。

2　寸法線位置を指示します。

3　直径記号と寸法値を確認して配置します。

プロパティエクスプローラー：

■ 円 or 円弧：直径を測る対象です。

■ 寸法線位置：直径寸法を置く場所です。

操作のポイント：

半径ではなく直径を表示します。穴寸法では円周を選択し、外径と内径を取り違えないようにします。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　要素間角度	コマンド ID：Constr.Angle.Dimension.Objects

要素（線分/円弧）の角度寸法を作図します

操作手順：

1　角度の基準になる第 1 要素を選択します。

2　第 2 の線分を選択します。

3　測る角の側に寸法線を配置します。

プロパティエクスプローラー：

■ 円弧または線分：第 1 要素を指定します。

■ 線分：第 2 要素を指定します。

■ 寸法線位置：角度を表示する側と位置を決めます。

操作のポイント：

内角か外側の角かで値が異なります。寸法線の円弧が目的の範囲を示すことを確認します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　3 点角度	コマンド ID：Constr.Angle.Dimension.3p

角度寸法（3 点）を作図します

操作手順：

1　角度の中心となる頂点を指示します。

2　第 1 辺の方向を決める点を指示します。

3　第 2 辺の方向を決める点を指示します。

4　寸法線位置を指定して配置します。

プロパティエクスプローラー：

■ 中心：角の頂点です。

■ 第 1 角度点・第 2 角度点：2 辺の方向を決めます。

■ 寸法線位置：角度寸法を置く場所です。

操作のポイント：

3 点は「頂点→第 1 辺上の点→第 2 辺上の点」の順に指定します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　4 点角度	コマンド ID：Constr.Angle.Dimension.4p

角度寸法（4 点）を作図します

操作手順：

1　第 1 辺上の 2 点を順に指示します。

2　第 2 辺上の 2 点を順に指示します。

3　寸法線の位置と測る角の範囲を確認して配置します。

プロパティエクスプローラー：

■ 第 1 点・第 2 点：第 1 辺の方向を決めます。

■ 第 3 点・第 4 点：第 2 辺の方向を決めます。

■ 寸法線位置：角度を表示する場所です。

操作のポイント：

2 辺の交点を直接指示しにくい場合に使用できます。各辺の 2 点を同じ位置にしないでください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　面取り寸法	コマンド ID：Constr.Chamfer.Dimension

45 度等距離の面取りに C 寸法を作成します

操作手順：

1　45 度の等距離面取り部分を選択します。

2　寸法文字を置く位置を指示します。

3　C 寸法として表示されることを確認します。

プロパティエクスプローラー：

■ 面取りエッジ：寸法を付ける面取り部分です。

■ 寸法位置：文字の配置位置です。

操作のポイント：

45 度等距離の面取り用です。角度や両側の距離が異なる場合は、長さ寸法と角度寸法で指示します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　引き出し線	コマンド ID：Constr.Labeling.Dimension

引き出し線を作図します

操作手順：

1　引き出し線の指示点を指定します。

2　折れ点を順に指示して文字位置へ導きます。

3　文字を入力し、線と文字が重ならない位置で確定します。

プロパティエクスプローラー：

■ 指示点：注記する対象の位置です。

■ 引き出し線の点列：必要な折れ点を指定します。

■ 文字：表示する注記を入力します。

■ モード：指示方法を切り替える場合に使用します。

タブメニュー：　ホーム	グループ：注釈
コマンド名：面にラベル貼付け	Command ID: MenuId.Constr.SurfaceLabel

面へ文字や画像をシールのように表示します。立体形状や体積は変更しません。
操作手順：
1　ホーム → 注釈 → 引き出し線▼ → 面にラベル貼付けを選び、貼付け面をクリックします。
2　「操作」を追加、「内容の種類」を文字または画像にして、文字列または画像ファイルを指定します。
3　幅・高さ・回転角・中心位置を調整し、プレビューを確認して右上の［✓］で確定します。
編集と削除：
貼付け面を選び、「対象ラベル」から対象を指定します。「操作」を編集または削除にして［✓］で確定します。面をクリックしただけでは終了しません。Undo で元へ戻せます。
設定項目：
フォント名は現在の文字フォントを最初に表示し、一覧から選択できます。幅・高さは mm、回転角は度で指定します。
平面・円筒・球面・自由曲面で使用できます。同じ曲面の連続面や継ぎ目をまたぎ、面の端と穴では切り抜きます。球面や自由曲面の端ではゆがみが出る場合があります。
ラベルは JDC へ埋め込まれ、移動・コピーに追従し、印刷と PNG に表示されます。STEP／STL にはラベルの見た目を出力しません。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　座標引き出し線	コマンド ID：Constr.Point.Dimension

点に座標寸法を作図します

操作手順：

1　座標を記入する点または対象要素を指示します。

2　座標文字を配置する位置を指示します。

3　表示される座標を確認して確定します。

プロパティエクスプローラー：

■ 指示点：座標を読み取る位置です。

■ 寸法線位置：座標注記の配置位置です。

■ モード：点の指定方法を切り替える場合に使用します。

操作のポイント：

表示位置と計測対象の点は別です。測りたい点をスナップで正確に指定してください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　バルーン	コマンド ID：Constr.Labeling.Balloon

バルーンコマンドを実行します。

操作手順：

1　記号設定で項目番号や接頭文字、尾文字を確認します。

2　矢印位置を指示します。

3　バルーンの円中心を指示して配置します。

プロパティエクスプローラー：

■ 矢印位置：注記する部品側の位置です。

■ 円中心：番号を囲む円の位置です。

■ 記号・接頭文字・尾文字：表示する内容です。

■ 連番方式：数値・アルファベット・連番なしを選択します。

操作のポイント：

部品表へ集計する場合は、項目番号と部品属性をそろえてください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　マーカー	コマンド ID：Constr.Labeling.Marker

マーカーコマンドを実行します。

操作手順：

1　記号設定で表示内容と連番方式を設定します。

2　マーカーの円中心を指示します。

3　番号と配置を確認して確定します。

プロパティエクスプローラー：

■ 円中心：マーカーを置く位置です。

■ 記号・接頭文字・尾文字：表示内容です。

■ 連番方式：番号の進め方を設定します。

操作のポイント：

バルーンとは異なり、矢印位置を指定せずに記号を配置します。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　幾何公差	コマンド ID：Constr.GeometricTolerance

幾何公差記入枠を作成します

操作手順：

1　幾何公差のダイアログで記号、公差値、必要なデータムを設定します。

2　設定内容を確定します。

3　図面上の配置点を指示して記入枠を置きます。

ダイアログと配置項目：

■ 公差記号・公差値：図面に指示する公差条件を設定します。

■ データム：参照する基準を設定します。

■ 配置点：記入枠を置く位置です。

操作のポイント：

公差値は設計条件に従って入力します。モデルから必要な公差を自動で判定する機能ではありません。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　データム	コマンド ID：Constr.DatumIdentifier

データム識別記号を作成します

操作手順：

1　データムを指示する面や位置に指示点を置きます。

2　識別記号を置く位置を指示します。

3　識別文字と指示先を確認します。

プロパティエクスプローラー：

■ 指示点：基準を示す位置です。

■ 記号位置：データムの枠を置く位置です。

操作のポイント：

自動設定された識別文字も確認し、幾何公差枠で参照する文字と一致させてください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　仕上げ記号	コマンド ID：Constr.SurfaceFinish

仕上げ記号コマンドを実行します。

操作手順：

1　仕上げ記号のダイアログで必要な条件を設定します。

2　指示点や折れ点を使う場合は、その位置を指定します。

3　記号位置を指示して配置します。

ダイアログと配置項目：

■ 記号の設定：仕上げ条件や表示内容を入力します。

■ 指示点・折れ点：引き出し線を付ける場合の位置です。

■ 記号位置：仕上げ記号の配置位置です。

操作のポイント：

モデル形状から表面粗さを測定する機能ではありません。指示する条件を入力してください。

タブメニュー：　ホーム	グループ：注釈
 コマンド名：　部品表	コマンド ID：Constr.Labeling.PartsList

部品表コマンドを実行します。

操作手順：

1　バルーンやマーカーの番号、部品属性を準備します。

2　部品表ダイアログを開き、集計内容と必要な項目を確認します。

3　名称や数量などを編集し、配置方向を設定します。

4　確定して図面上の配置位置を指示します。

部品表ダイアログ：

■ 項目・部品番号・名称・数量：表の内容を確認・編集します。

■ 配置方向：上から追加／下から追加を選択します。

■ CSV 書き出し：編集中の表を CSV へ出力します。

操作のポイント：

作成済みの表を選んで同じコマンドを実行すると編集できます。部品や番号を変更した後は再集計結果を確認します。

タブメニュー：　挿入	グループ：挿入
 コマンド名：　ブロック挿入	コマンド ID：Insert

ブロックもしくは図面を挿入します

操作手順：

1　挿入元を Block またはファイルから選択します。

2　ブロック名または JDC ファイルを指定します。

3　挿入点、尺度、回転角を設定して配置します。属性入力が出た場合は値を入力します。

プロパティエクスプローラー：

■ 挿入元・Block・ファイル：挿入するデータを選びます。

■ 挿入点：ブロックの基点を置く位置です。

■ X 尺度・Y 尺度・Z 尺度：各方向の倍率です。

■ 回転角：挿入時の向きです。

■ 分解して挿入：初回は「しない」。前回の選択を引き継ぎます。

操作のポイント：

尺度 1 で実寸を保ちます。登録済みブロックは定義の基点、ファイルは図面の挿入基点を使います。「する」は外側の一階層を分解します。詳しくは第 3 部を参照してください。

タブメニュー：　挿入	グループ：挿入
 コマンド名：　CAD 図形挿入	コマンド ID：Import

JDC、DWG、DXF、STEP の CAD 図形を追加挿入します

操作手順：

1　挿入する CAD ファイルを選択します。

2　読み込んだ形状を確認し、配置操作が表示されたら挿入位置を指定します。

3　現在の図面へ追加された位置と大きさを確認します。

操作のポイント：

図面の原点を基準に挿入し、元のブロック参照と対応するダイナミック情報を保持します。図面全体を一つのブロックにする場合は「ブロック挿入」を使います。

タブメニュー：　挿入	グループ：挿入
 コマンド名：　SVG 図形挿入	コマンド ID：Import.SVG

SVG 図形を追加挿入します

操作手順：

1　挿入する SVG ファイルを選択します。

2　取り込む図形を確認し、必要な配置指定を行います。

3　線や曲線の位置、大きさ、閉じた輪郭を確認します。

操作のポイント：

SVG の表現によっては CAD の線・塗りに同じ形で変換されない場合があります。加工用に使う輪郭は閉図形チェックでも確認します。

タブメニュー：　挿入	グループ：挿入
 コマンド名：　ブロック	コマンド ID：BlockManager

フォルダライブラリから JDC ブロックをプレビューして挿入します

操作手順：

1　ライブラリのタブとフォルダを選択します。

2　一覧から JDC ブロックを選び、プレビューを確認します。

3　尺度、角度、分解挿入の設定を確認して挿入します。

4　図面上で配置位置を指定します。

ブロックマネージャー：

■ ライブラリタブ・フォルダ：使用するブロックの保存場所です。

■ プレビュー：挿入前に形状を確認します。

■ 尺度・角度：配置時の大きさと向きです。

■ 分解挿入：まとまりを維持するか、構成要素として挿入するかを選びます。

操作のポイント：

タブの削除は登録を外す操作で、保存フォルダのファイルを削除する操作ではありません。

タブメニュー：　挿入	グループ：ブロック定義
 コマンド名：　属性定義	コマンド ID： Constr.Block.AttributeDefinition

ブロックに含める属性を作成します

操作手順：

1　タグ、プロンプト、規定値を入力します。

2　文字高さ、フォント、表示の設定を行います。

3　挿入点を指示して属性定義を配置します。

4　部品図形と属性定義を一緒に選び、ブロックを作成します。

プロパティエクスプローラー：

■ タグ：属性を識別する名前です。

■ プロンプト・規定値：挿入時の案内と初期値です。

■ 挿入点・文字高さ・フォント：属性文字の位置と書式です。

■ 非表示・定数など：属性の表示や入力方法を指定します。

操作のポイント：

属性定義をブロックに含めると、挿入した部品ごとに番号や名称などを入力できます。

タブメニュー：　挿入	グループ：ブロック定義
 コマンド名：　ブロック作成	コマンド ID：Block

選択オブジェクトから名前付きブロック定義を作成し、ブロック参照に置き換えます

操作手順：

1　ブロックにまとめる図形と、必要な属性定義を選択します。

2　ブロック名を入力します。

3　挿入時の基準にする基点を指示して確定します。

プロパティエクスプローラー：

■ 名前：図面内で使用するブロック名です。

■ 基点：挿入や配置の基準位置です。

■ 部品属性：必要な場合に設定します。

操作のポイント：

選択した図形はブロック参照に置き換わります。繰り返し使用する部品の管理に向いています。

タブメニュー：　挿入	グループ：ブロック定義
 コマンド名：　ブロック書き出し	コマンド ID：WBlock

選択図形の座標を維持し、指定点を図面の挿入基点として JDC へ保存します

操作手順：

1　ファイルへ書き出す図形を選択します。

2　ブロックの基点を指示します。

3　保存先と名前を指定して JDC ファイルを書き出します。

プロパティエクスプローラー：

■ 基点：出力図面に保存する挿入基点です。未設定の図面は原点を使用します。

■ 部品属性：必要な場合に設定します。

操作のポイント：

基点を部品の取付点や中心にすると、別の図面へ挿入するときに位置合わせしやすくなります。

タブメニュー：　挿入	グループ：書き出し
 コマンド名：　CAD 図形書き出し	コマンド ID：Export.Selection

選択した要素を書き出します

操作手順：

1　出力する CAD 要素を選択します。

2　保存形式、保存先、ファイル名を指定します。

3　書き出しを実行し、必要に応じて出力先 CAD で形状を確認します。

操作のポイント：

形式やライセンスによって使える出力が異なります。ソリッド、曲線、色など、受け渡したい情報を保持できる形式を選んでください。

タブメニュー：　挿入	グループ：書き出し
 コマンド名：　SVG 図形書き出し	コマンド ID：Export.SVG

選択した 2D 要素を SVG ファイルへ書き出します

操作手順：

1　SVG へ出力する 2D 要素を選択します。

2　保存先とファイル名を指定して書き出します。

3　SVG を開き、範囲や線の表示を確認します。

操作のポイント：

2D 図形の交換用です。3D ソリッドをそのまま保持する形式ではありません。

タブメニュー：　挿入	グループ：書き出し
 コマンド名：　レイアウトを 2D 書き出し	コマンド ID：LayoutView.Export2D

現在のレイアウトを重複整理した 2D 図面として JDC、DWG、DXF、JWW へ書き出します

操作手順：

1　出力するレイアウトをアクティブにします。

2　コマンドを実行し、JDC・DWG・DXF・JWW などの保存形式と保存先を指定します。

3　書き出した 2D 図面で、ビューと寸法の配置を確認します。

操作のポイント：

レイアウトを 2D 要素へ変換した出力です。元モデルとの関連を保持した編集用データとしては、元の JDC 図面も保存してください。

タブメニュー：　挿入	グループ：参照
 コマンド名：　アタッチ	コマンド ID：ExternalReference.Attach

外部参照図面を読み込みます

操作手順：

1　参照する図面ファイルを選択します。

2　参照方法やファイルパスの設定を確認します。

3　挿入位置を指定して参照図面を配置します。

操作のポイント：

外部参照は参照元ファイルを利用する仕組みです。参照元を移動・改名した場合は参照管理でパスを確認してください。

タブメニュー：　管理/計測	グループ：管理
 コマンド名：　名前削除	コマンド ID：Purge

ブロック定義や画層など使用されていない項目を図面から削除

操作手順：

1　名前削除ダイアログを開きます。

2　未使用のブロック定義・レイヤなどから削除する項目を選びます。

3　対象を確認して削除を実行します。

操作のポイント：

図形を直接削除するコマンドではありません。後で使うために登録した未使用の定義も候補になるため、残す項目を確認してください。

タブメニュー：　管理/計測	グループ：管理
 コマンド名：　参照管理	コマンド ID：ExternalReference.Manager

現在挿入されている外部参照図形を管理します

操作手順：

1　参照管理を開き、参照名と状態を確認します。

2　更新した参照元を反映する場合は再読み込みを行います。

3　見つからない参照はパスを修正し、不要な参照は解除します。

参照管理の主な操作：

■ 読み込み・アンロード：配置を残したまま参照の表示を切り替えます。

■ 再読み込み：参照元の変更を反映します。

■ パス変更：参照ファイルの場所を変更します。

■ 解除：その参照の配置と定義を取り除きます。

操作のポイント：

解除はアンロードと異なり、配置も削除します。同じ参照を複数配置している場合は、すべてへの影響を確認してください。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　2 点間距離	コマンド ID：Measure.Calculate.Distance3D

3 次元空間の 2 点間の距離を計測します

操作手順：

1　距離を測る 1 点目を指示します。

2　2 点目を指示して計算結果を確認します。

3　結果を図面へ残す場合は文字位置を指示して確定します。

プロパティエクスプローラー：

■ 1 点目・2 点目：測定位置です。

■ 計算結果：3 次元空間での距離が表示されます。

■ 文字位置：計測結果の注記を置く場所です。

操作のポイント：

画面への投影距離ではありません。高さが異なる 2 点では Z 方向の差も含めて計算します。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　UCS 角度	コマンド ID：Measure.Calculate.UcsAngle

2 点を結ぶ方向を現在の UCS へ投影し、UCS X 軸基準の角度を 0～360° で計算します。結果文字を配置します。

操作手順：

1　角度の基準にしたい UCS を設定しておきます。

2　方向を決める 1 点目と 2 点目を指示します。

3　計算結果を確認し、必要なら文字位置を指定します。

プロパティエクスプローラー：

■ 1 点目・2 点目：1 点目から 2 点目へ向かう方向を決めます。

■ 計算結果：UCS へ投影した方向の角度です。

■ 文字位置：結果を記入する位置です。

操作のポイント：

UCS の X 軸を基準に 0～360° で表示します。2 点の順序を逆にすると方向も反対になります。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　面の角度	コマンド ID：Measure.Calculate.FaceAngle

選択面と現在の UCS 平面との傾きを計算します。結果文字を配置します。

操作手順：

1　比較の基準となる UCS を設定します。

2　角度を測る面を選択します。

3　計算結果を確認し、必要なら結果文字を配置します。

プロパティエクスプローラー：

■ 面：角度を測る対象面です。

■ 計算結果：面と UCS 平面との傾きが表示されます。

■ 文字位置：結果を図面へ残す位置です。

操作のポイント：

2 面を選んで二面角を測る操作ではありません。現在の UCS 平面を基準とする 0～90° の角度です。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　閉図形の周長	コマンド ID：Measure.Calculate.Perimeter

閉じた曲線を現在の UCS へ投影し、その周長を計算します。結果文字を配置します。

操作手順：

1　計算の基準となる UCS を設定します。

2　円や閉じたパス、または端点でつながった線分・円弧の輪郭をクリックします。

3　周長を確認し、必要なら文字位置を指定して確定します。 認識した輪郭全体のハイライトが意図した範囲か確認します。

プロパティエクスプローラー：

■ 閉図形：周長を求める閉じた曲線です。

■ 計算結果：UCS へ投影した輪郭の長さです。

■ 文字位置：結果を配置する位置です。

操作のポイント：

接続した線分・円弧は先に結合せず計測できます。隙間・分岐・非表示要素を含む輪郭は認識できません。

Esc で計測プレビューを解除します。UCS による投影と軸の基準を計算前に確認してください。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　面積・断面特性	コマンド ID： Measure.Calculate.SectionProperties

閉図形について、面積、図心、X 軸・Y 軸断面二次モーメントを計算します。結果文字を配置します。

操作手順：

1　断面の基準にする UCS を設定します。

2　円や閉じたパス、または端点でつながった線分・円弧の輪郭をクリックします。

3　面積、図心、断面二次モーメントを確認します。 認識した輪郭全体のハイライトが意図した範囲か確認します。

4　必要なら結果文字を配置します。

プロパティエクスプローラー：

■ 閉図形：同じ平面で閉じた輪郭を指定します。

■ 計算結果：面積、図心、X 軸・Y 軸の断面二次モーメントです。

■ 文字位置：計算結果の注記を置く位置です。

操作のポイント：

接続した線分・円弧は先に結合せず計測できます。隙間・分岐・非表示要素を含む輪郭は認識できません。

Esc で計測プレビューを解除します。UCS による投影と軸の基準を計算前に確認してください。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　面と面の距離	コマンド ID： Measure.Calculate.FaceDistance

選択した 2 面間の最短距離を計算し、最短点を結ぶ線を作成します。現状は 0.01 mm 精度の近似計算です。

操作手順：

1　距離を求める 1 面目を選択します。

2　2 面目を選択し、計算結果と最短点を結ぶ線を確認します。

3　確定すると、最短点間を結ぶ線が作成されます。

プロパティエクスプローラー：

■ 1 面目・2 面目：最短距離を測る面です。

■ 計算結果：面間の距離を表示します。

操作のポイント：

曲面の三角形化に基づく近似計算です。0.01 mm は計算に使う精度設定であり、あらゆる曲面で結果の誤差が 0.01 mm 以内になることを保証する値ではありません。

タブメニュー：　管理/計測	グループ：計測
 コマンド名：　ソリッド容積・重心	コマンド ID： Measure.Calculate.SolidMassProperties

1 個または複数ソリッドの合計容積と重心座標を計算し、重心点を作成します。質量・密度の計算は含みません。

操作手順：

1　容積と重心を求めるソリッドを選択します。複数選択もできます。

2　合計容積と重心座標を確認します。

3　確定すると、重心位置に点を作成します。

プロパティエクスプローラー：

■ ソリッド：計算対象となる閉じた立体です。

■ 計算結果：合計容積と体積で加重した重心座標です。

操作のポイント：

材質や密度を入力する質量計算ではありません。複数ソリッドが重なる場合は各容積の合計となるため、重複を除いた容積が必要なら先に和演算してください。

操作場所： レイアウト	グループ：レイアウト
レイアウト 表示とビューの管理	コマンド ID：LayoutView / LayoutPatch

操作の入口

プロパティエクスプローラーの「ビュー」タブで、レイアウト名またはその下のビュー名を右クリックします。レイアウト名と個々のビューでは、操作の対象が異なります。

レイアウトを開く

1　レイアウト名のメニュー先頭にある「表示」を選ぶと、その用紙へ切り替わります。

2　同じグループの「名前変更」「削除」は、そのレイアウトを対象にします。削除前に対象名を確認してください。

ビューの表示とロック

個々のビューの「ビューを表示」をオフにすると、画面・印刷・2D 書き出しから除外します。ビューはツリーに残るため、そこから再表示できます。「すべてのビューを再表示」でまとめて戻せます。

配置を固定する

「ビューをロック」は用紙上の位置・枠・尺度・投影方向を固定します。ロック中も図形へのスナップと用紙上の寸法・文字の編集は可能です。用紙全体のズームやパンもできます。

新規レイアウトビュー

「新規レイアウトビュー」で通常のビューを追加できます。三面図など投影方向の関係を持たせたい場合は、次項の VIEWBASE と VIEWPROJ を使用します。

操作のポイント

ビューを非表示にしても、用紙へ手動で描いた独立した文字や線は残ります。「内側移動をロック」は従来の枠と内容の移動方法の設定で、「ビューをロック」とは働きが異なります。

操作場所： レイアウト	グループ：レイアウト
レイアウト VIEWBASE ベースビュー	コマンド ID：ViewBase

概要

モデルを用紙へ配置し、三面図などの基準となる投影ビューを作成します。モデルそのものの寸法や向きを変更する操作ではありません。

操作手順

1　モデルを用意し、配置するレイアウトを表示します。

2　レイアウトの右クリックメニューから「VIEWBASE（ベースビュー）」を選びます。

3　方向・尺度・表示方法・投影法を設定します。

4　用紙上へマウスを動かして配置位置を決め、クリックして確定します。

主な入力項目

方向は正面・上面・右側面などから選びます。尺度は実寸に対する倍率で、1 は 1:1、0.1 は 1:10、2 は 2:1 です。表示方法は隠れ線などの見え方、投影法は第一角法・第三角法の配置関係を指定します。

配置後の操作

ビューを選択すると、プロパティで尺度などを確認できます。関連する別方向の図は「投影ビューを追加（VIEWPROJ)」で追加します。用紙のサイズとビューの尺度は別々に設定します。

操作のポイント

ビューは元のモデルを参照します。レイアウトへ直接描いた文字・寸法などは用紙上の要素として扱います。配置を整えた後は「ビューをロック」を使うと意図しない移動を防げます。

操作場所： レイアウト	グループ：レイアウト
レイアウト 投影ビューを追加	コマンド ID：ViewProject

概要

基準ビューから、位置と投影方向の関係を持つ別方向のビューを追加します。

操作手順

1　レイアウトで基準となる投影ビューを 1 つ選択します。

2　右クリック→「投影ビューを追加（VIEWPROJ)」を選びます。

3　基準ビューの上下左右、または斜めへマウスを動かします。配置方向に対応した投影候補を確認します。

4　目的の位置でクリックして確定します。必要な図は同じ操作で追加します。

引き継ぐ設定

基準ビューのモデル、尺度、表示方法、投影法を使用します。追加操作中はこれらの項目を基準ビューから継承するため、個別に変更できない項目があります。

投影法の確認

上下左右のどちらにどの面が現れるかは、基準ビューの第一角法・第三角法に従います。配置前にプレビューで向きを確認してください。

操作のポイント

基準には対応する平行投影ビューを選びます。透視ビューや断面ビューは対象外です。面を正面から見た図が必要な場合は、次項の補助投影ビューを使用します。

操作場所： レイアウト	グループ：レイアウト
レイアウト 面に正対する補助投影ビュー	コマンド ID：ViewAuxiliary

概要

斜めの平面を正面から見る投影図を追加します。斜面上の穴や輪郭を、その面に正対した向きで確認できます。

操作手順

1　基準の VIEWBASE または VIEWPROJ のビューを 1 つ選びます。

2　右クリック→「面に正対する補助投影ビュー」を選びます。

3　ビュー内の平面をクリックします。面が横から見て線になっている場合は、そのエッジをクリックします。

4　「見る側」と「配置方法」を確認し、用紙上の配置位置をクリックします。

入力項目

「基準の平面／エッジ」では候補の面を確認できます。「見る側」は表側と反対側を切り替えます。「配置方法」は「投影方向に位置合わせ」または「自由配置」を選びます。

作成結果

元ビューのモデル、尺度、レイヤ表示、可視線・隠線の設定を継承します。指定した面だけを切り抜く図ではなく、その方向からモデル全体を投影します。

操作のポイント

対象は平面です。曲面、透視ビュー、断面ビューからは作成できません。投影方向は作成時に決まるため、後から基準面の傾きを変えた場合は向きを確認してください。Esc で中止できます。

操作場所： レイアウト	グループ：レイアウト
レイアウト 円形の詳細図を追加	コマンド ID：ViewDetail

概要

元ビューの一部を円形の範囲で取り出し、拡大した詳細図を配置します。

操作手順

1　基準となる投影図または補助投影図を 1 つ選びます。

2　右クリック→「円形の詳細図を追加」を選びます。

3　元ビュー内で範囲の中心をクリックし、半径をクリックまたは紙面上の mm で入力します。

4　必要に応じて「尺度係数（実寸基準)」を変更し、元の範囲から離れた位置へ配置します。

尺度と注記

初期尺度は元ビューの 3 倍です。元が 1:10 なら 3:10、1:1 なら 3:1 になります。入力値 3 は実寸に対して 3:1 です。元の範囲記号、詳細図の円枠・タイトル・尺度が自動作成されます。

詳細図名の変更

詳細図を選び、プロパティの「詳細図名（記号)」を変更します。右クリックの「名前変更」からも変更できます。元の範囲記号とタイトルが連動します。同じレイアウト内で重複する名前や空欄は使用できません。

操作のポイント

寸法は拡大倍率ではなくモデルの実寸を示します。詳細図は移動と尺度変更が可能です。断面・透視・詳細図を元にした再詳細化は対象外です。円の範囲は作成時の投影位置で保持されるため、モデル変更後は範囲を確認してください。

操作場所： レイアウト	グループ：レイアウト
レイアウト 断面ビュー	コマンド ID：ViewSection

概要

投影図に切断線を指定し、その位置で切断した断面図を用紙へ配置します。元のモデルを切り離す操作ではありません。

操作手順

1　レイアウトで基準となる投影図を選択します。

2　右クリック→「断面ビュー」を選びます。

3　切断線を 2 点以上で入力し、Enter で切断線を決定します。

4　マウスで断面図の配置位置を決め、矢印と見る方向を確認して確定します。Esc で中止できます。

断面の種類

1 区間の切断線は直線断面です。平行な複数の切断区間を接続した折れ線ではオフセット断面を作成します。非平行な切断区間では、同一断面へ回転して展開する整列断面として扱います。

作成される表示

切断位置の線、見る方向の矢印、識別文字と A-A などの断面名、切断面のハッチングを作成します。配置側によって見る方向が変わるので、矢印を必ず確認してください。

操作のポイント

切断線は元ビューの中で指定します。形状を更新した後は切断位置と断面の内容を確認してください。図面として残す線・文字と、モデルを切断するモデリングコマンドを区別して使用します。

操作場所： レイアウト	グループ：レイアウト
レイアウト 稜線を非表示と再表示	コマンド ID：LayoutPatch.HideDrawingEdges

概要

指定ビュー内の不要な稜線・投影輪郭を非表示にして、図面の見え方を整えます。元のモデル形状や別のビューには影響しません。

操作手順

1　編集するレイアウトビューを選択します。

2　右クリック→「稜線を非表示」を選びます。

3　非表示にする線をクリック、または矩形範囲で選択します。可視線だけでなく、表示されている隠れ線も選べます。

4　Enter で確定します。Esc は取消です。

非表示を戻す

ビューの右クリックメニューから「非表示稜線をすべて再表示」を実行すると、そのビューの個別の非表示設定を解除できます。直前の操作は「元に戻す」でも戻せます。

表示の違い

この操作は選んだ線だけを隠します。ビュー全体を消す場合は「ビューを表示」、隠れ線を含む表示方式を変える場合はビューの表示設定を使います。

操作のポイント

用紙に直接描いた独立した線は、ビューの稜線とは別の要素です。非表示設定は JDC に保存され、印刷や 2D 書き出しにも反映されます。モデルの形状を大きく変更した後は仕上がりを確認してください。

操作場所： レイアウト	グループ：レイアウト
レイアウト 稜線の太さを変更	コマンド ID：LayoutPatch.SetEdgeWidth

概要

レイアウトのビュー全体、または選んだ稜線だけに、紙面上の線幅を設定します。元モデルの線幅属性は変更しません。

ビュー全体の線幅

1　対象ビューを選択し、プロパティの「稜線の太さ（mm)」へ数値を入力します。

2　0 を指定すると、元のモデルの線幅属性を使用します。

選んだ稜線の線幅

1　プロパティの「選択稜線の太さ（mm)」へ適用する値を入力します。

2　その項目または右クリックメニューから「選択稜線の太さを変更」を実行します。

3　対象の稜線をクリックまたは矩形で選択し、Enter で確定します。Esc で中止できます。

設定の優先順位

個別の稜線設定はビュー全体の設定より優先されます。個別設定に 0 を適用すると、その稜線をビュー全体の設定に戻します。

尺度との関係

数値は紙面上の mm です。ビューを 1:10 から 1:20 へ変更しても指定した紙面線幅を維持します。レイアウト画面自体のズームでは、用紙と同じ倍率で線幅が表示されます。

操作のポイント

設定は JDC へ保存され、印刷や 2D 書き出しにも反映されます。確定後は印刷プレビューで他の線とのバランスを確認してください。

第 3 部 ダイナミックブロック

Free でも 2D ダイナミックブロックの作成・編集と、完成済み 3D ブロックの操作を利用できます。3D 面動作の作成・編集と DWG／DXF 出力は STD 以上です。再編集用原本は JDC へ保存します。

パラメーターは「変更する値や方向」、動作は「どの図形をどう動かすか」です。距離パラメーターを置くだけでは図形は伸びません。ストレッチなどの動作を関連付けてください。

この部の読み方

初めて作る場合は 01〜07 を順に操作します。連動は 10・18、3D 面動作・数式・可視性の詳細は 24〜33 を参照してください。

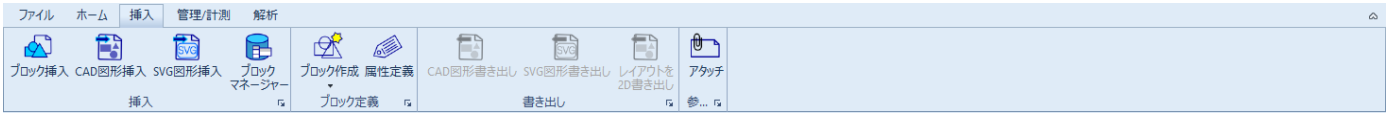


図 3-A ［挿入］タブの基本操作例。ブロック定義のメニューから作成を開始します。

01 配置済みブロックを操作する 278
02 作成と編集を開始する 279
03 パラメーターと動作を選ぶ 280
04 距離パラメーターを作る 281
05 ストレッチの伸縮範囲を指定する 282
06 動かす図形を選び確定する 283
07 動作テストと保存 284
08 スナップとトラッキング 285
09 回転と反転を設定する 286
10 回転に距離パラメーターを追従させる 287
11 可視状態を切り替える 288
12 XY と極方向を設定する 289
13 尺度を設定する 290
14 配列を設定する 291
15 ルックアップ表を作る 292
16 設定を後から変更する 293
17 基点と位置合わせ 294
18 チェーンアクションを設定する 295
19 チェーンが動かないときの確認 296
20 挿入と分解と WBLOCK 297
21 DWG と DXF へ動作付きで保存する 298
22 現在の形状で保存される設定 299
23 FREE で使う機械部品のダイナミックブロック 300

図形・名称・数値は操作例です。次の詳細目次から、3D 面動作・数式・可視性・受け渡しの説明へ移動できます。

第 3 部 詳細目次：3D・規格表・運用

01〜23 の基本操作に続き、3D 部品の作成と受け渡しを説明します。

24 3D ダイナミックブロックの設計 301
25 実習：端面を動かして長さを変える 302
26 面のオフセット・回転を登録する 303
27 穴径変更と穴・突起のコピー 304
28 複数ソリッドの面を一つの動作で動かす 305
29 ルックアップ表で数式を使う 306
30 規格名の判定と表のメンテナンス 307
31 可視状態とグリップ表示をそろえる 308
32 保存形式と受け渡しの判断 309
33 ダイナミックブロックの確認表 310

01 配置済みブロックを操作する

対応したブロックを選択すると、グリップと、右側の［ダイナミックブロック］項目が表示されます。距離・角度は数値入力、可視状態やルックアップは選択肢から変更できます。

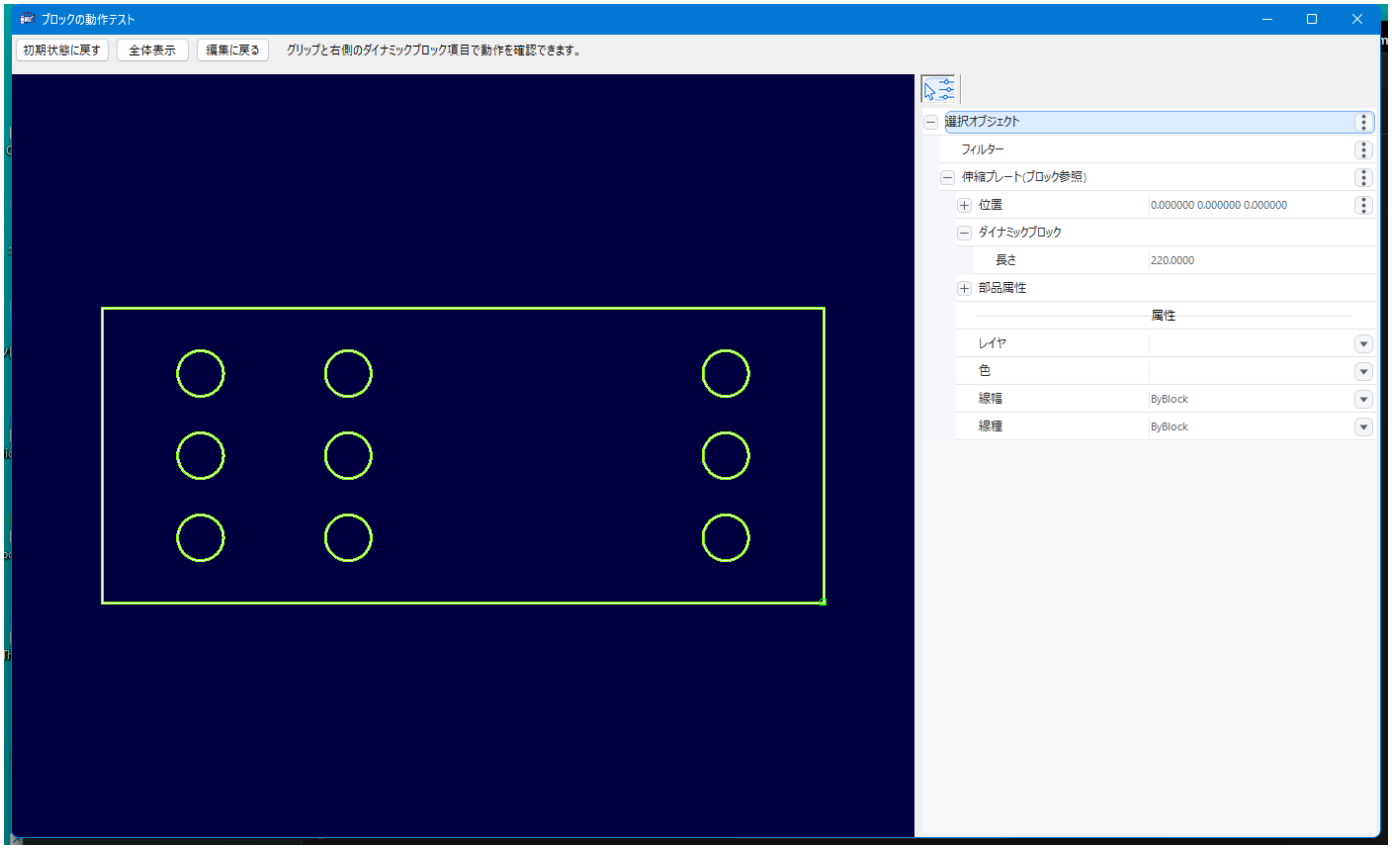


図 3-1 動作テストで長さを変更した例。右側のダイナミックブロック項目で値を確認します。

1. ブロックをクリックし、右側のパラメーター名を確認します。
2. 長さや角度を入力して確定するか、対象のグリップをドラッグします。
3. 変更を戻す場合は［元に戻す］を使います。定義の初期値へ戻す操作とは区別してください。

読み込んだ AutoCAD ブロック

変更済みの匿名ブロックは、外観の保持を優先して読み込む場合があります。右クリックに［初期状態に戻してダイナミック編集を開始］が出る場合、その操作で配置が初期化されます。現在の寸法・形状を残す必要があれば、先に JDC で別名保存してください。

読み取れない設定や拘束を含む定義は、すべてを編集できるとは限りません。通常のブロックと同じ見え方でも、対応する動作の有無を確認してください。

02 作成と編集を開始する

新規作成は、元の図形を選択して［挿入］タブの［ブロック定義］グループで、［ブロック作成］の▼→［ダイナミックブロックを作成…］を選びます。未選択から始め、エディター内で作図することもできます。

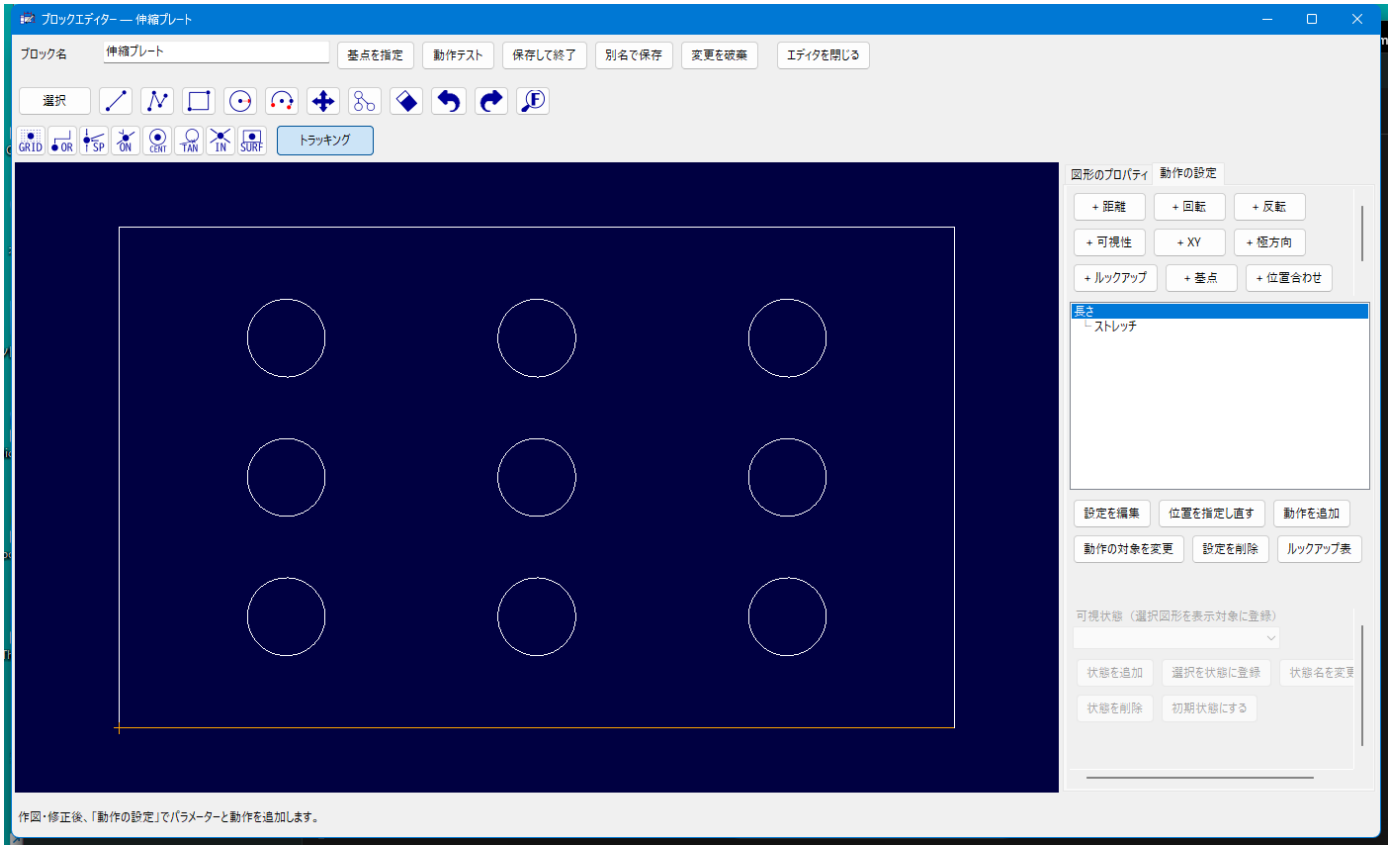


図 3-2 ブロックエディターの画面例。上部で作図と保存、右側でパラメーターと動作を設定します。

既存の編集は、ブロックを選択して右クリック→［ブロック定義を編集…］です。配置を選択していない場合は、名前付き定義の一覧から選びます。

画面の役割

［図形のプロパティ］では図形の属性、［動作の設定］では距離などのパラメーターと関連する動作を設定します。編集画面では、可視性で隠す図形も含めて表示します。実際の見え方は［動作テスト］で確認してください。

編集画面は元の図面とは別の作業領域です。元の図面への反映は［保存して終了］で行い、その後に図面ファイルも保存します。

03 パラメーターと動作を選ぶ

目的に合うパラメーターを追加します。2D の動作はブロックの XY 平面を基本とし、3D 面動作は XYZ の方向・軸を設定します。詳細は 24 以降を参照してください。

パラメーター	主な用途	関連する操作
距離	長さや間隔を変える	移動、ストレッチ、均等尺度、配列
回転	角度を変える	回転
反転	左右などを反転する	反転
可視性	形状の表示を切り替える	状態ごとの図形を登録
XY	横と縦の範囲を変える	移動、ストレッチ、配列
極方向	距離と方向を変える	極ストレッチ、移動、回転、尺度、配列
ルックアップ	規格名から複数値を選ぶ	数値表を登録
基点	挿入基準を指定し追従させる	他の動作への追従を設定
位置合わせ	線や円弧へ向きを合わせる	専用の動作追加は不要
位置	1 点の位置を変える	移動などと連動

共通の設定

［表示名］は配置後のプロパティに表示する名前です。［最小値］［最大値］［刻み］［候補値］で値を制限できます。候補値は「100;150;200」のように半角セミコロンで区切り、制限しない欄は空欄にします。

［プロパティに表示］と［グリップを表示］は別の設定です。数値からだけ操作させたい場合は、グリップの表示をオフにします。

04 距離パラメーターを作る

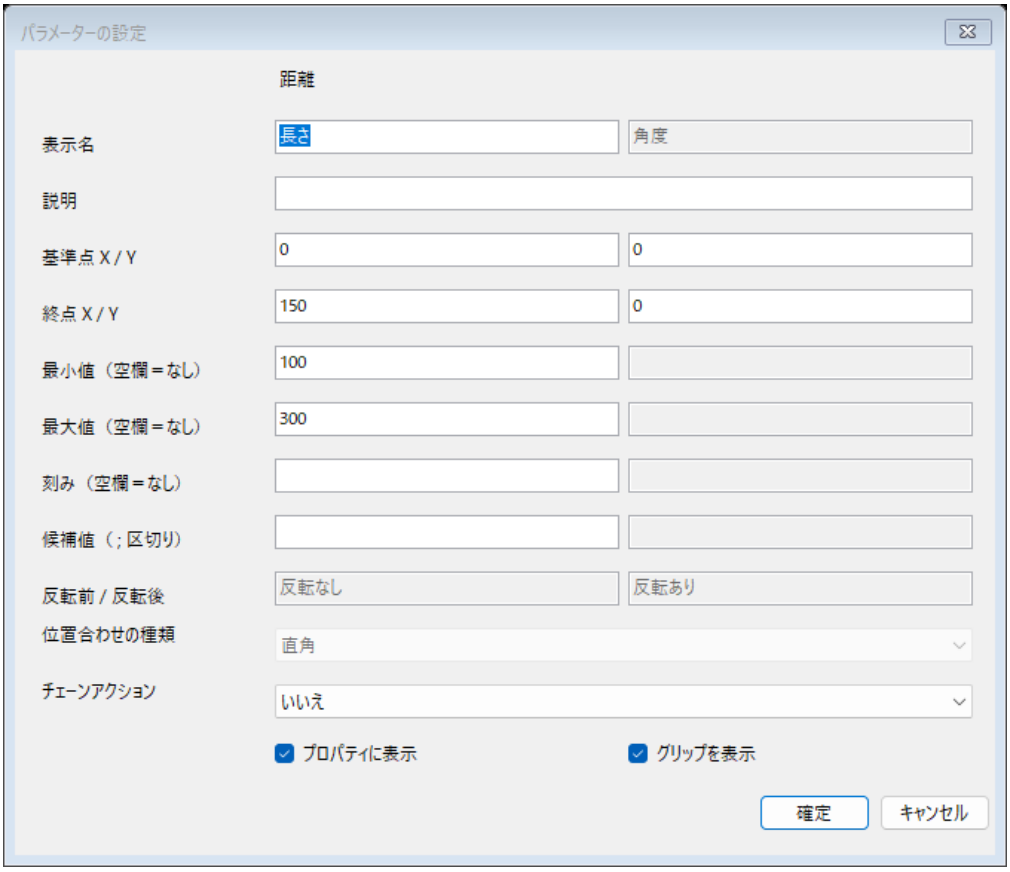


図 3-3 距離パラメーターの設定。単純なストレッチではチェーンアクションは「いいえ」のままで構いません。

1. ［動作の設定］→［＋距離］を押します。
2. 基準点、終点の順に 2 点をクリックします。円の中心間や矩形の端点間など、変更する寸法に合う点を指定します。
3. 表示名を「長さ」などにし、必要な制限値を入力して［確定］します。
4. 右側の一覧に作成したパラメーターが表示されたことを確認します。

図では長さ 150 を基準とし、最小 100、最大 300 に制限しています。チェーンアクションは、別のパラメーターから連動を受ける場合に「はい」にします。

05 ストレッチの伸縮範囲を指定する

1. 右側の一覧で距離パラメーターを選択します。図形上のオレンジ色の線は、選択したパラメーターの位置表示です。
2. ［動作を追加］→［ストレッチ］を選び、設定を確定します。
3. 伸縮範囲の対角 2 点をクリックします。右端を伸ばす例では、移動する右端だけを枠に含め、固定する左端を枠の外にします。

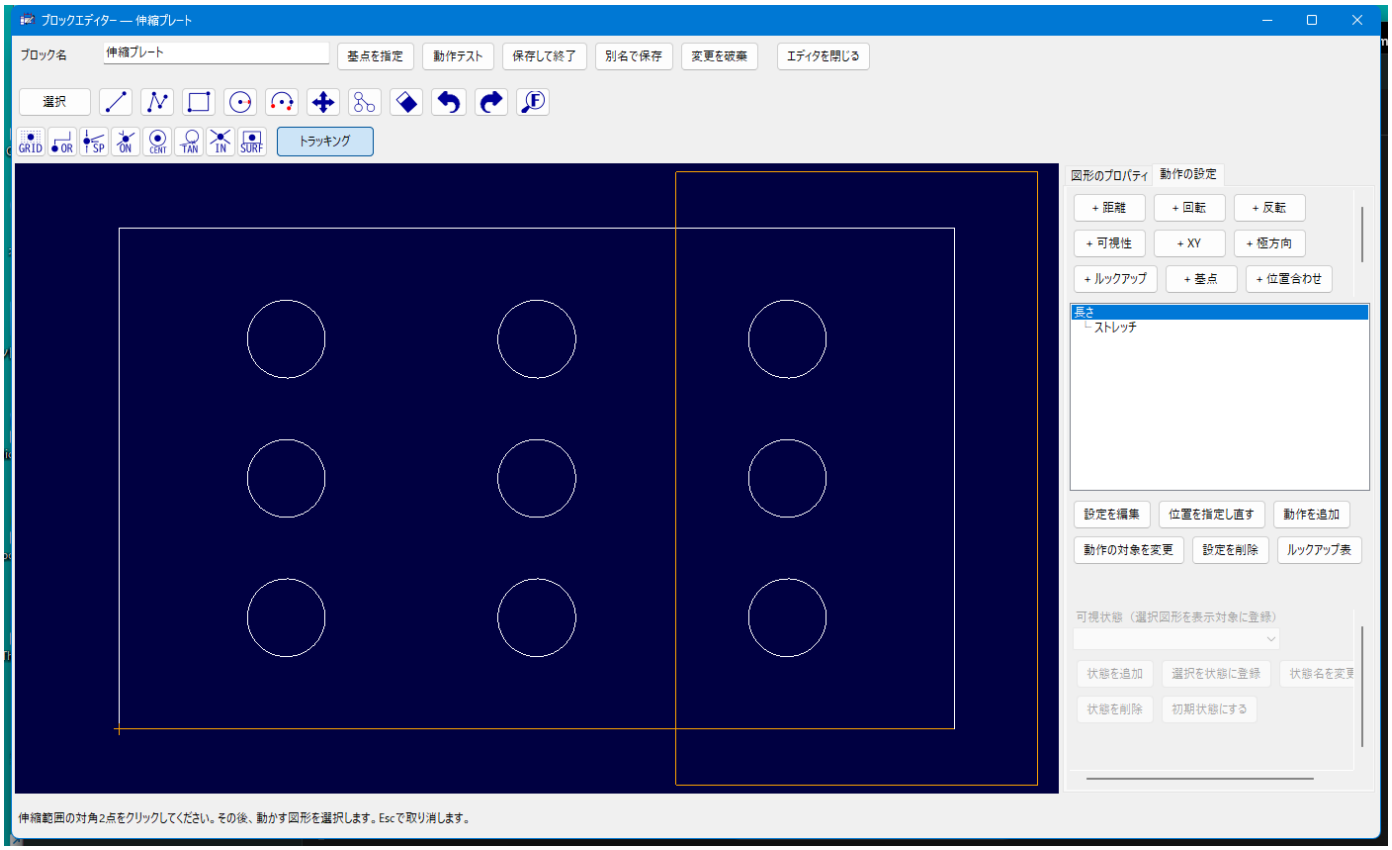


図 3-4 オレンジ色の伸縮枠を指定した例。枠を作った後に、動作対象の図形を選びます。

枠と対象図形の違い

枠は「どの端点を動かすか」、次に行う対象選択は「どの図形に作用させるか」を指定します。枠だけを作っても登録は完了しません。

線分・直線ポリラインの枠内の端点が伸縮します。図形全体が枠に入れば全体が移動します。円などの曲線全体の移動は可能ですが、曲線をまたぐ部分ストレッチは対象外です。

06 動かす図形を選び確定する

伸縮枠を指定した後、ステータスバーの案内が対象選択に変わったら、動かす図形をクリック、またはドラッグで囲みます。

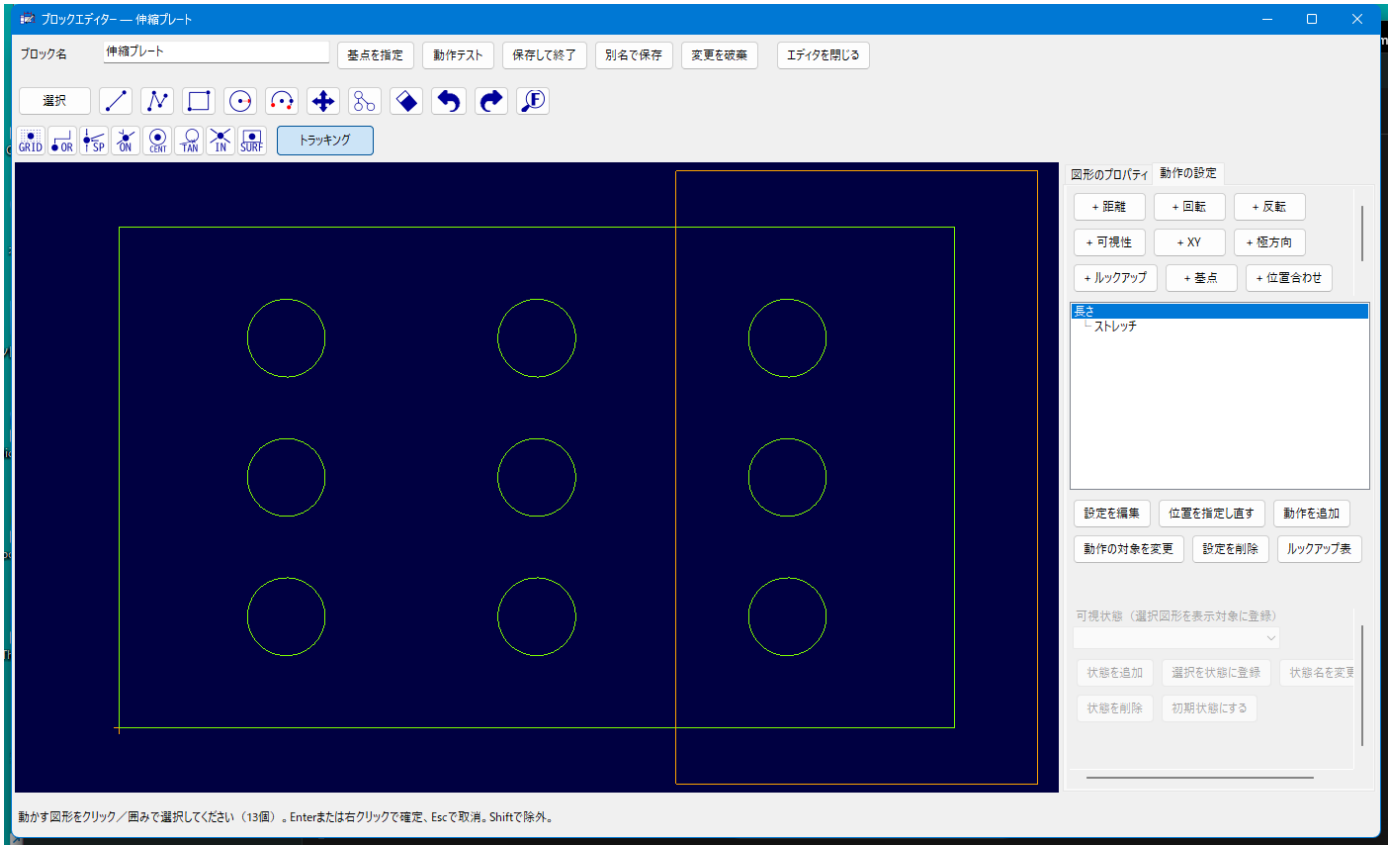


図 3-5 対象図形を選択する画面。全体を選んでも、伸縮枠の外側の端点は固定されます。

1. 右から左へ囲むと、枠に触れる図形も選択します。左から右へ囲むと、完全に入った図形を選択します。
2. 選択済みの図形をクリックすると除外します。Shift を押しながら囲むとまとめて除外できます。
3. 選択数を確認し、Enter または右クリックで確定します。Esc は取り消しです。

単純なストレッチでは、距離パラメーターをこの段階で選び直す必要はありません。別のパラメーターも連動させるときは、動作設定画面の［対象パラメーター］で指定します。

画面下部の対象数と、Enter または右クリックで確定する案内を確認してください。

07 動作テストと保存

［動作テスト］は、元の図面に反映する前にグリップと数値入力を試す画面です。

1. ［動作テスト］を押し、対象ブロックを選択します。
2. グリップをドラッグするか、右側のダイナミックブロック項目で値を変更します。
3. ［初期状態に戻す］でテスト中の変更を戻せます。［編集に戻る］で設定を続けます。
4. 設定ができたら［保存して終了］を押し、元の CAD 画面で図面を保存します。



図 3-6 エディター上部の保存・終了ボタン。動作テストもここから開始します。

ボタン	反映する内容
保存して終了	定義を保存して閉じる。同じ定義を使う配置にも反映
別名で保存	別の定義を登録。元の配置は置き換えない
変更を破棄	確認後、編集内容を反映せず閉じる
エディタを閉じる または ×	変更があれば保存確認。キャンセルで編集を続行

配置ごとの値は、変更後の定義でも有効なら保持します。候補値の削除などで保持できない場合は、保存前の案内を確認してください。元の図面で［元に戻す］を実行すると、定義と配置の変更をまとめて戻します。

08 スナップとトラッキング

点を指定するときは、上部のオブジェクトスナップを使います。端点、中心・中点、交点などを必要に応じて切り替え、取得位置のマークを確認してクリックします。



図 3-7 スナップとトラッキングの操作欄。各スナップを必要に応じて切り替えます。

パラメーターを配置するとき

距離・XY・極方向などの 1 点目からスナップを利用できます。2 点目の指定中も設定を切り替えられます。［トラッキング］を有効にし、基準にする点へカーソルを置いて取得してから、目的の方向へ動かします。

スナップの設定はエディター起動時の CAD 設定を引き継ぎ、閉じると元の CAD の設定に戻ります。点が取れない場合は、対象に合う種類がオンか、取得マークが表示されているかを確認してください。

配置後のグリップを動かすとき

距離・回転のグリップは、通常の極トラッキング設定と角度間隔を利用します。回転中心や距離の基準点を基に補助線が表示されます。確定すると補助線は消え、Esc でドラッグを取り消します。

矩形やダイナミックブロックのグリップ変更は、ドラッグ確定を単位として元に戻します。

09 回転と反転を設定する

回転

［＋回転］で中心点と基準方向の点を指定し、表示名・角度の制限を設定します。一覧で回転パラメーターを選び、［動作を追加］から回転を設定して、回転させる図形を選びます。角度と刻みは度で入力します。

反転

［＋反転］で反転軸の 2 点を指定し、反転前・反転後の状態名を設定します。反転動作を追加して対象図形を登録し、動作テストで表裏の状態を確認します。

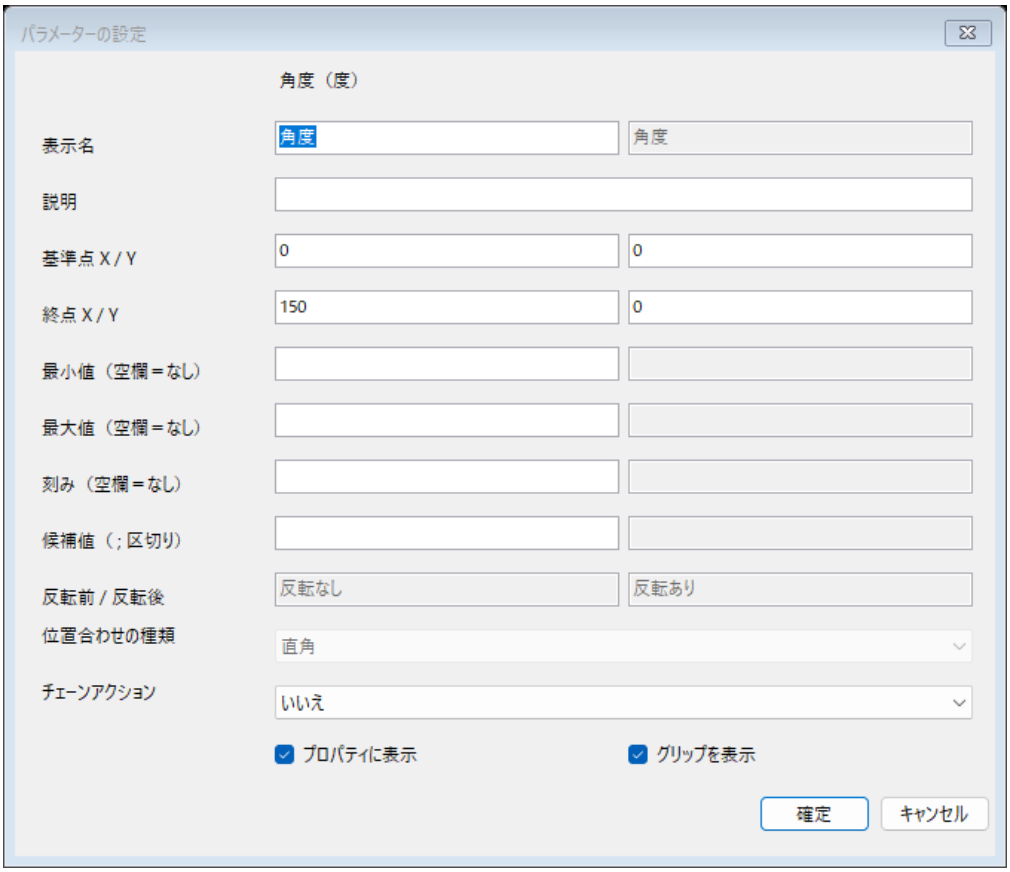


図 3-8 回転パラメーターの設定。基準点と終点から回転中心と基準方向を決めます。

パラメーターの位置は［位置を指定し直す］、名称・座標・制限は［設定を編集］で変更します。図形全体を回したい場合と、一部だけ回したい場合で、動作対象を分けてください。

10 回転に距離パラメーターを追従させる

図形だけを回転対象にすると、距離パラメーターの軸は元の方向に残ります。回転後の斜めの方向へ長さを変更するには、距離パラメーターも回転動作の対象に含めます。

1. 距離パラメーターにストレッチを設定します。
2. 回転パラメーターの動作を一覧で選択し、［設定を編集］を開きます。
3. ［対象パラメーター］で距離パラメーターにチェックします。回転させる図形も回転動作に登録します。
4. 動作テストで角度を変更し、距離のグリップが図形と同じ方向を向くことを確認します。
5. その状態で距離を変更し、回転後の軸に沿って伸縮することを確認します。

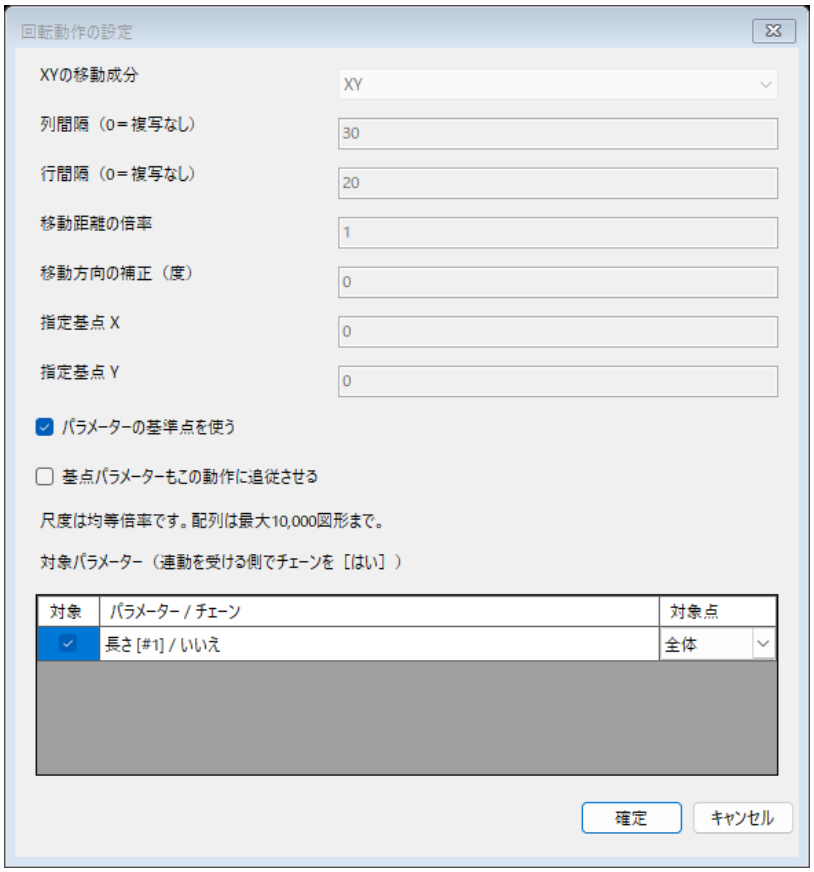


図 3-9 回転動作で、距離パラメーターを対象に含める設定。

回転によって距離の始点と終点が同じ角度だけ回る場合、長さそのものは変わりません。位置や方向を追従させる設定と、値を変えて子の動作を実行するチェーンアクションは区別します。

11 可視状態を切り替える

可視性は、一つのブロックに用意した複数の形を「閉」「半開」「開」などの名前で切り替える機能です。

1. 各状態に使う図形を作成し、［＋可視性］で切替グリップの位置を指定します。
2. 一覧で可視性パラメーターを選択します。
3. ある状態で表示する図形を選択し、［状態を追加］で状態名を入力します。
4. 選択を変えて他の状態も登録します。共通で表示する図形は各状態に含めます。
5. ［動作テスト］で状態を切り替えて確認します。

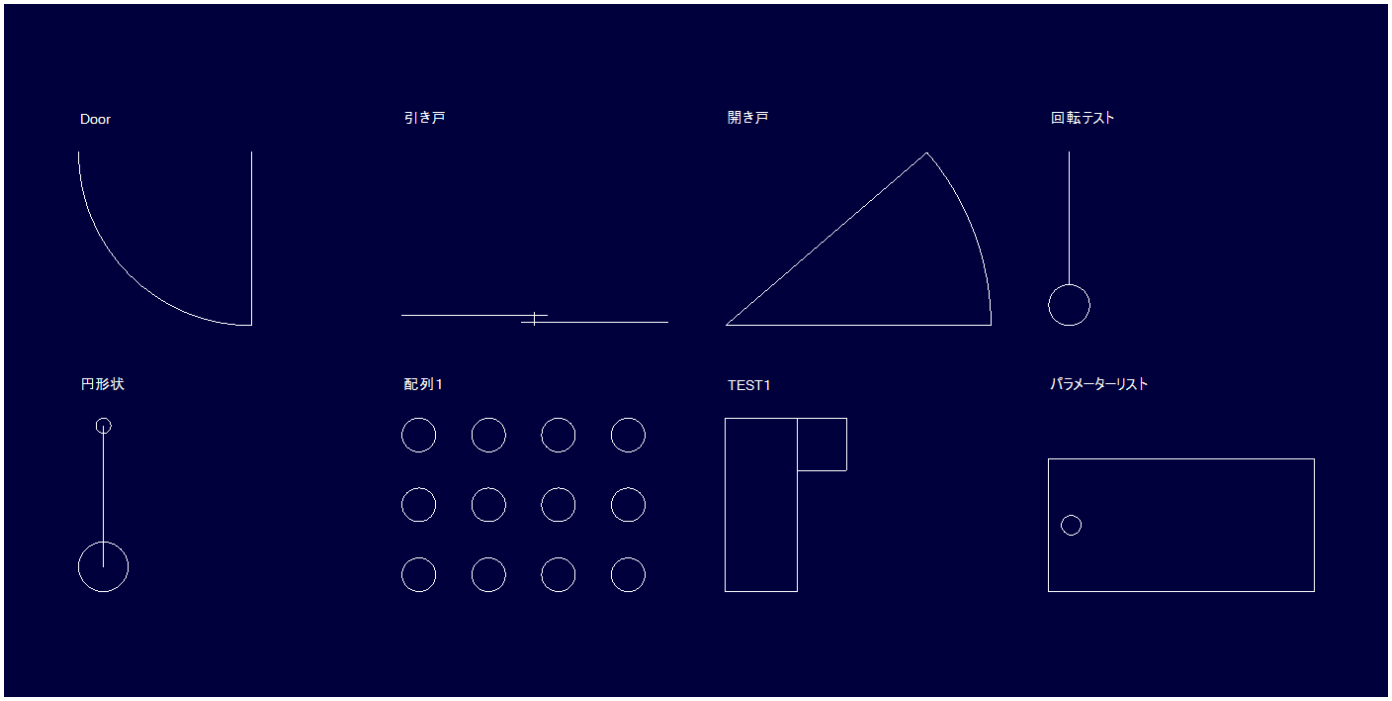


図 3-10 ダイナミックブロックを読み込んだ表示例。可視性に対応する定義では、その状態に含まれる図形を表示します。

既存状態は、図形を選び直して［選択を状態に登録］で更新します。名前変更、削除、［初期状態にする］も使えます。可視性パラメーターは一つで、移動や回転の動作追加は不要です。

選択なしで状態を登録すると、その状態では管理対象の図形をすべて隠します。エディターは全図形を表示するため、状態の見え方は動作テストで確認してください。

12 XY と極方向を設定する

XY

［＋XY］で基準点と終点を指定し、X と Y の表示名・制限値を設定します。設定画面の左右の欄が X と Y に対応します。

移動・ストレッチの動作設定で［XY の移動成分］を X、Y、XY から選びます。横幅を変える図形を X、高さを変える図形を Y の動作へ別々に登録すると、一つのパラメーターで幅と高さを操作できます。

極方向

［＋極方向］で基準点と終点を指定し、距離と角度を設定します。角度は度です。極ストレッチでは、矩形内の端点を距離と角度の変化に合わせて動かします。

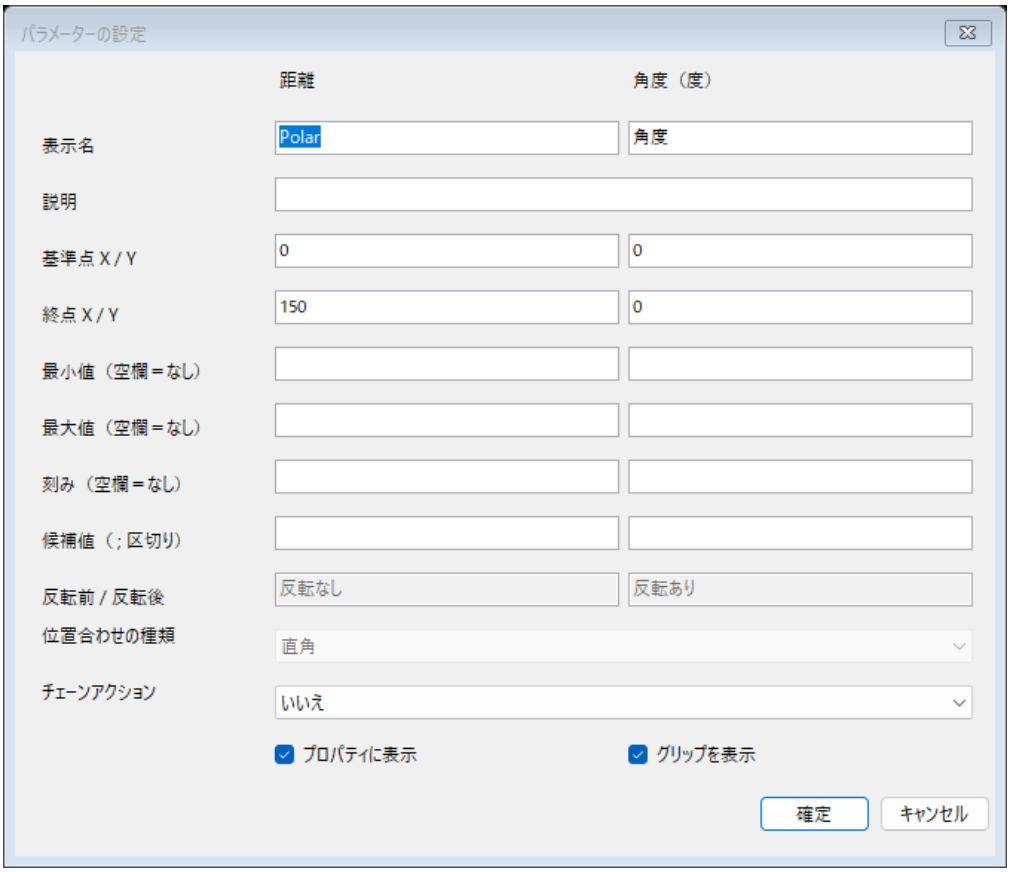


図 3-11 極方向の距離と角度を設定する画面例。

矩形内に図形全体が入れば全体が移動します。AutoCAD の極ストレッチにある回転専用選択セットは対象外です。負の XY 距離など、JDC で操作できても DWG・DXF で動作付き保存の対象外になる設定があります。

13 尺度を設定する

距離パラメーターに尺度動作を設定すると、初期距離との比率で対象図形を均等に拡大・縮小します。

1. ［＋距離］で基準となる 2 点を指定します。初期距離を 100 とする場合は、100 離れた 2 点を使います。
2. パラメーターを一覧で選び、［動作を追加］→［尺度］を選びます。
3. 基点を確認します。パラメーターの基準点を使うか、指定基点 X・Y を設定します。
4. 拡大・縮小する図形を選び、Enter または右クリックで確定します。
5. 動作テストで距離を 200 にすると 2 倍、50 にすると 0.5 倍になります。

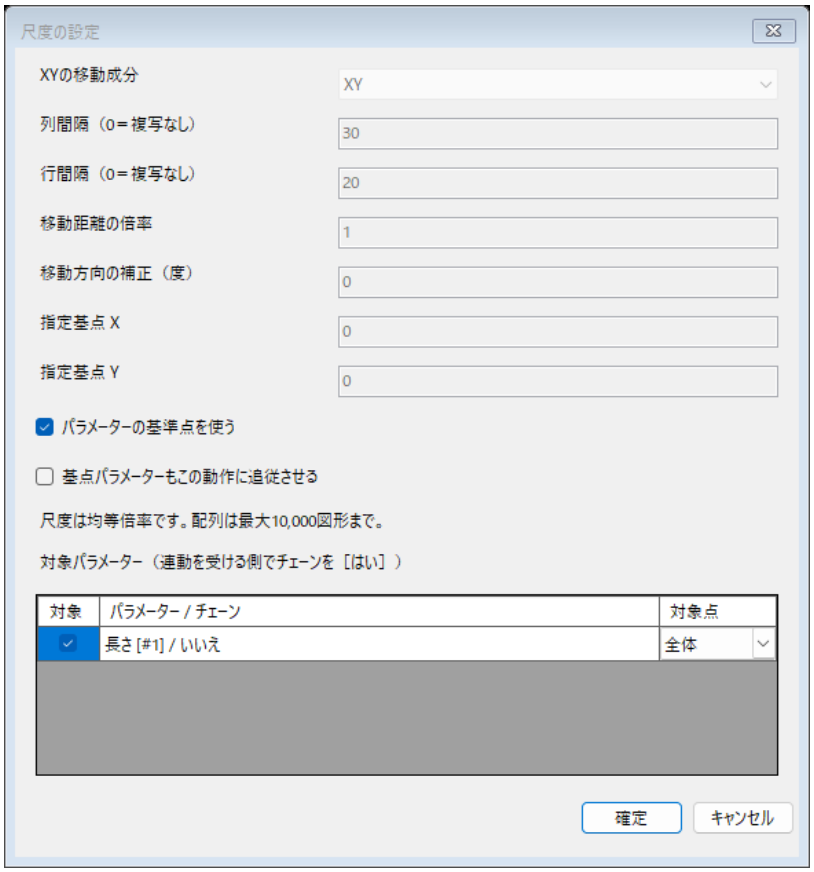


図 3-12 尺度の基点と対象パラメーターを指定する画面。

基点と対象の確認

基点は拡大・縮小しても動かない基準です。取付位置を固定したまま大きくしたい場合は、その取付位置を基点にします。

他のパラメーターも同じ比率で追従させる場合は、動作設定の対象パラメーターへ登録します。基点パラメーターを追従させる設定は、必要な場合だけ有効にしてください。

均等尺度に対応します。X と Y で別々の倍率を掛ける非均等尺度アクションには対応していません。

DWG・DXF へ動作付きで保存する場合は、AutoCAD 2018 以降の形式を選び、保存後の案内も確認してください。

14 配列を設定する

配列は、距離や XY の範囲に応じて同じ図形を一定間隔で増減する動作です。穴やボルトなど、繰り返す図形を対象にします。

1. 距離・XY・極方向のいずれかのパラメーターを作成します。
2. ［動作を追加］→［配列］で列間隔・行間隔を指定します。
3. 繰り返す元の図形を選び、Enter または右クリックで確定します。
4. 動作テストで範囲を変え、列数・行数と位置を確認します。

設定	複写の方向と数
距離または極方向	列間隔を指定。行間隔は 0
XY	列間隔と行間隔を指定。0 の方向には複写しない
個数の計算	元図形を含め、各方向で範囲÷間隔の整数部分＋1
例	X＝30、Y＝20、間隔が各 10 なら 4 列×3 行

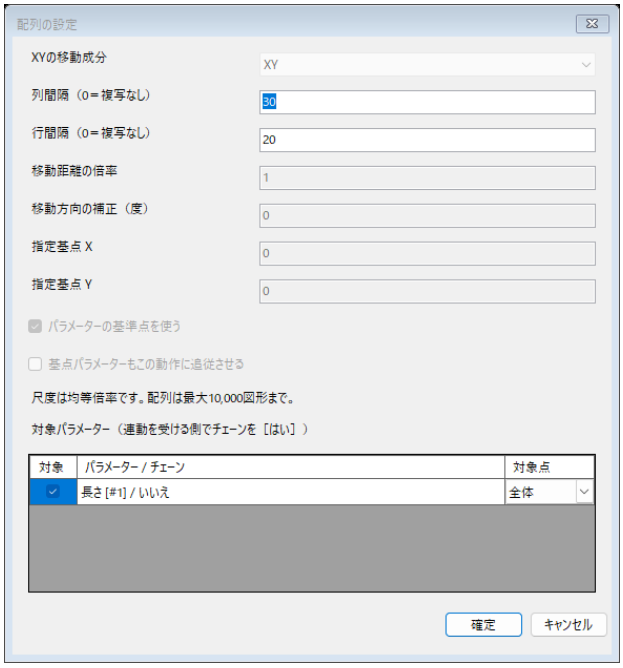


図 3-13 XY 配列の動作設定。列間隔と行間隔を入力します。

生成後の合計が 10,000 図形を超える変更は実行しません。DWG・DXF では、負の XY 距離や片方の間隔が 0 の XY 配列などは現在形状で保存する場合があります。

15 ルックアップ表を作る

規格名を選ぶだけで、複数の距離や角度を一緒に変更できます。先に寸法パラメーターと、それに対応する動作を登録してください。

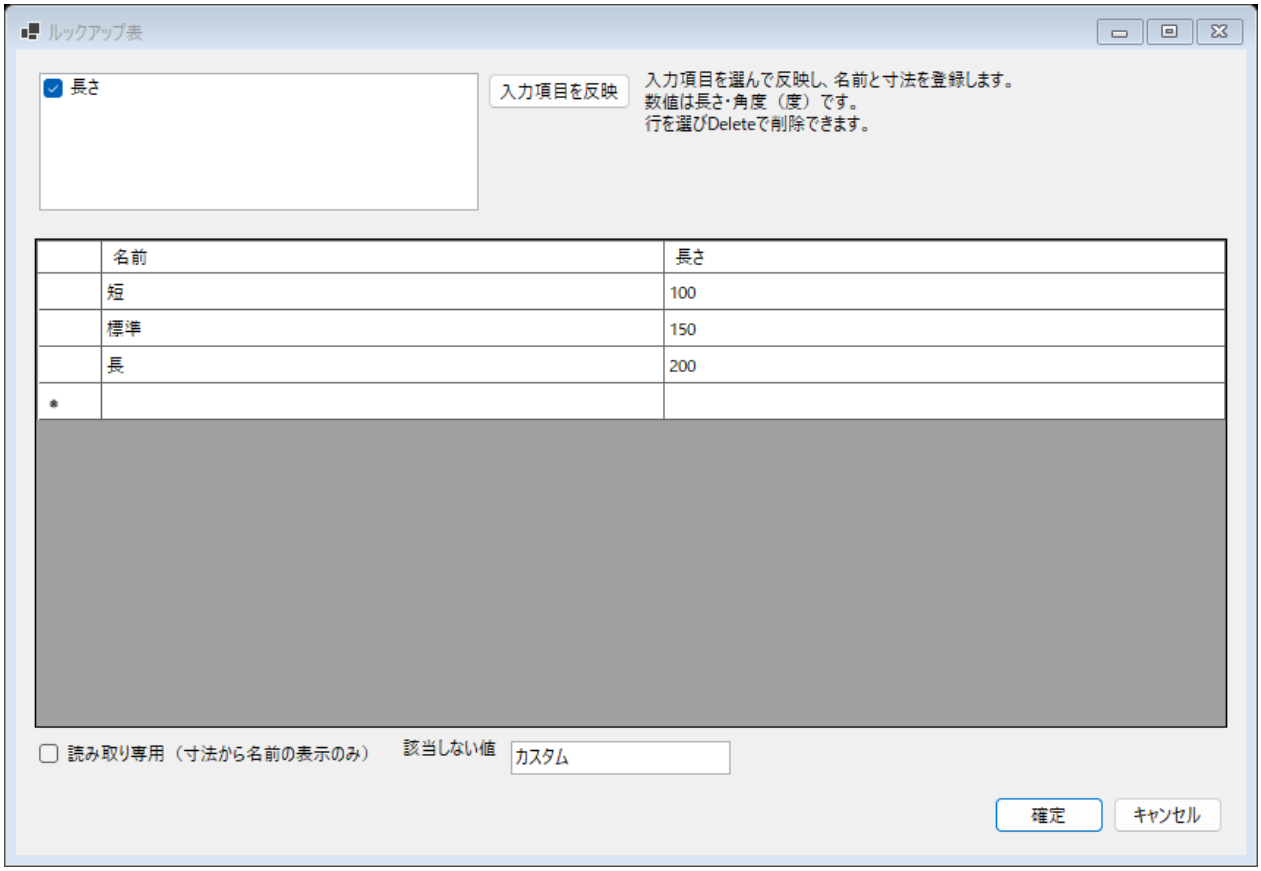


図 3-14 「短」「標準」「長」に長さ 100、150、200 を登録した例。

1. ［＋ルックアップ］でグリップ位置と表示名を設定します。
2. 表の上部で入力項目にチェックし、［入力項目を反映］を押します。
3. 各行に名前と対応する寸法・角度を入力します。角度は度です。
4. ［確定］し、動作テストで名前を選びます。

数値を直接変更した場合も、一致する行があればその名前を表示します。一致しない場合は「カスタム」などを表示します。［読み取り専用］では寸法から名前を表示する用途になり、名前を選んで寸法を変える操作はできません。

候補や制限値に反する数値、選択名の重複、判定が曖昧な重複行は登録できません。数値の完全一致による表を扱い、範囲・ワイルドカード・複数出力の表は対象外です。

16 設定を後から変更する

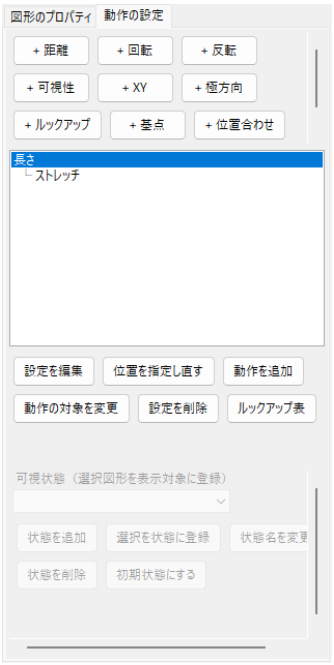


図 3-15 右側の一覧と編集ボタン。パラメーターとその下の動作を区別して選択します。

変更したい内容	操作
表示名、座標、制限値、チェーン	パラメーターを選び［設定を編集］
間隔、倍率、基点、連動する対象	動作を選び［設定を編集］
動かす図形	動作を選び［動作の対象を変更］し、選択を確定
パラメーターの配置位置	［位置を指定し直す］
ルックアップの規格と寸法	パラメーターを選び［ルックアップ表］

ルックアップは、一覧で選択すると表示されるオレンジ色の四角をドラッグして位置を変えられます。マウスを離して確定し、Esc で取り消します。スナップも利用できます。

ルックアップ表の行を変更しても、配置済みの有効な寸法は自動で置き換えません。新しい規格寸法を適用する場合は、配置側で名前を選び直してください。

17 基点と位置合わせ

挿入基点と動作の基点

上部の［基点を指定］は挿入基準を変更します。基点パラメーターがある場合はその位置を、ない場合は通常の挿入基点を変更します。

［＋基点］は、他の動作に追従させることもできる基点パラメーターを一つ追加します。距離変更に応じて挿入基準も移動させる場合は、動作設定の［基点パラメーターもこの動作に追従させる］を使います。

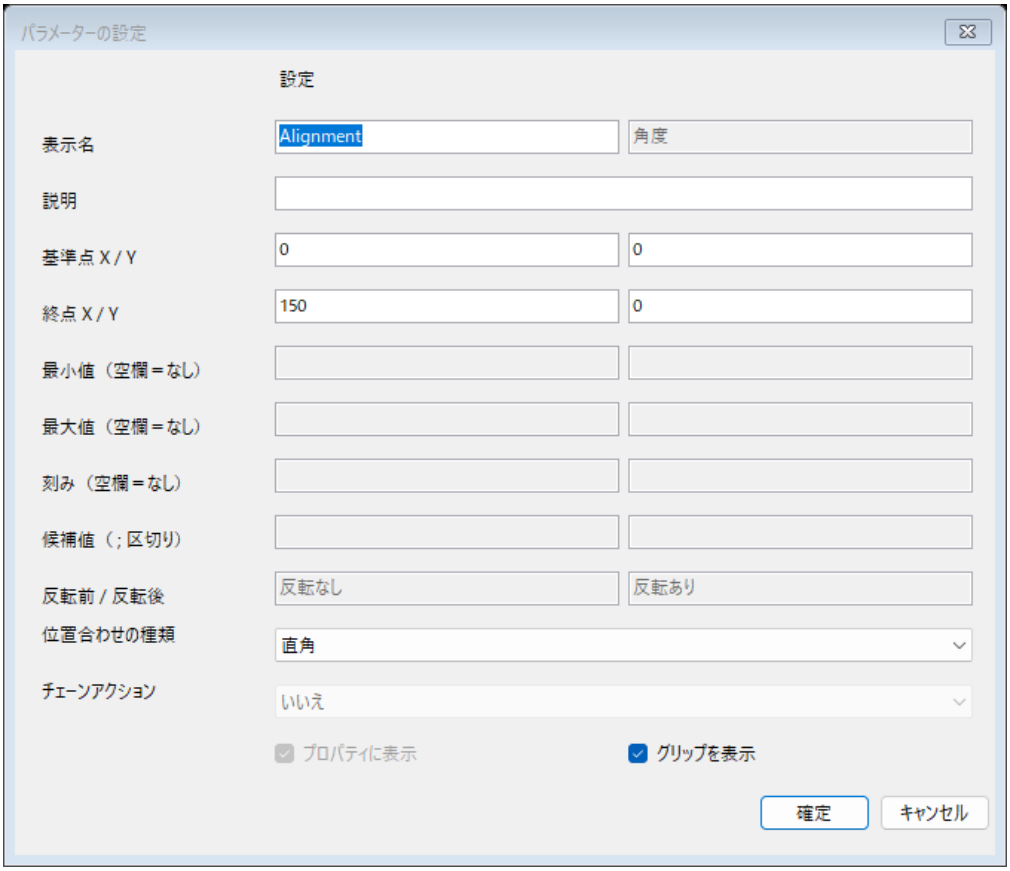


図 3-16 位置合わせの設定。直角と接線方向から選択します。

位置合わせ

［＋位置合わせ］で基準点と方向の 2 点を指定します。挿入時または位置合わせグリップのドラッグ時に、同じ平面上の線分・円弧・円などへ近づけると位置と向きが合います。Ctrl を押すと反対向きになります。

位置合わせはブロック全体を動かすため、別の動作追加は不要です。任意の 3D 面やスプラインへの自動位置合わせは対象外です。位置合わせを含む定義の DWG・DXF 保存は、現在形状になる制限があります。

18 チェーンアクションを設定する

チェーンアクションは、あるパラメーターの変更を別のパラメーターへ伝え、その先の動作も実行する設定です。

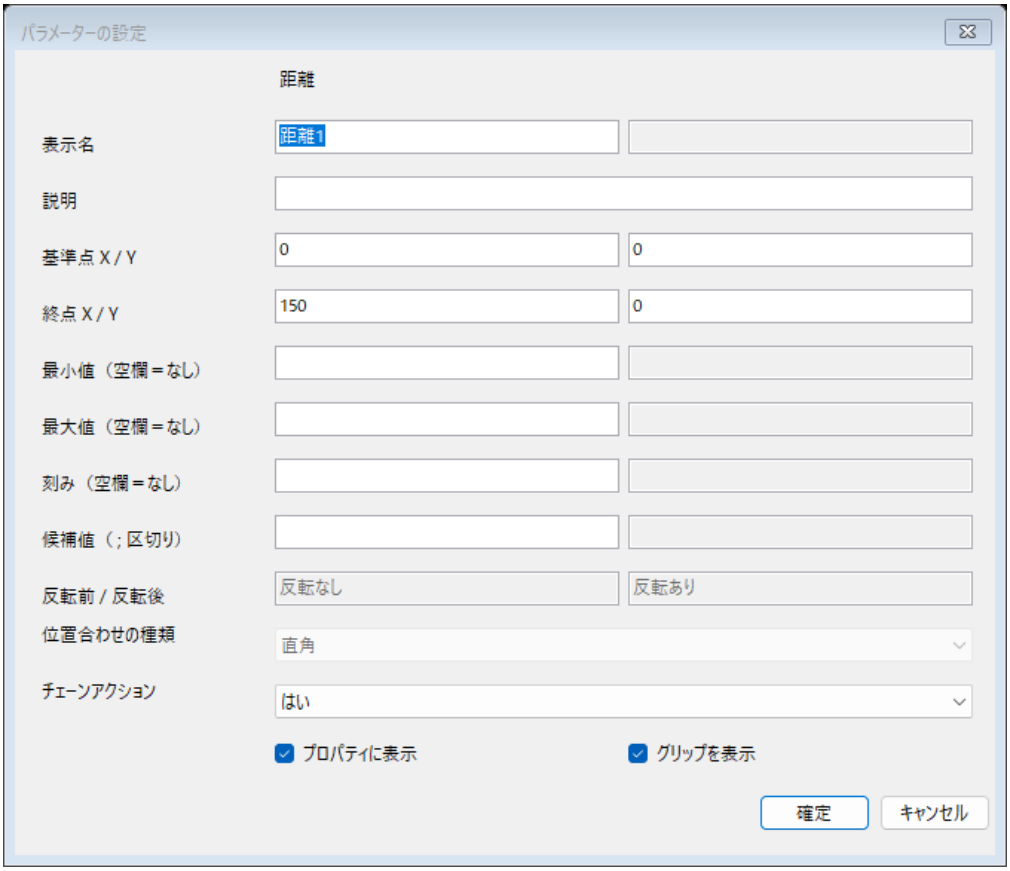


図 3-17 連動を受ける側でチェーンアクションを「はい」にします。

1. 連動を受ける側の「距離 1」を選び、［設定を編集］で［チェーンアクション］を［はい］にします。
2. 距離 1 には、それで動かす図形とストレッチなどの動作を登録しておきます。
3. 連動を始める「距離 2」の動作設定を開き、対象パラメーターで距離 1 にチェックします。
4. 距離 1 の長さを変える場合、ストレッチの対象点は通常［終点］にします。
5. 動作テストで距離 2 を変更し、距離 1 とその対象図形が変わることを確認します。

19 チェーンが動かないときの確認

症状	確認する設定
距離 1 が移動するだけで長さが変わらない	対象点が［全体］になっていないか。長さ変更には始点または終点を指定
距離 1 の位置や値は変わるが図形が動かない	連動を受ける距離 1 のチェーンが［はい］か。距離 1 自身の動作と図形が登録されているか
図形が二重に動く	同じ図形を親と子の両方の対象に入れていないか
変更が取り消される	子の最小値・最大値・刻みなどに違反していないか
設定が登録できない	自分自身や一周する連動になっていないか

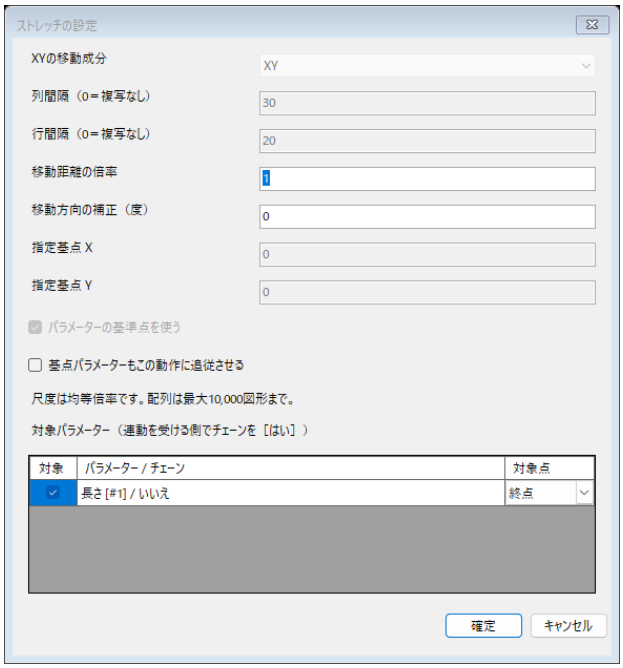


図 3-18 対象パラメーターと対象点を指定する欄。位置パラメーターは 1 点なので「全体」を使います。

パラメーター全体の平行移動では、始点と終点の間隔は変わりません。「位置が追従すること」と「距離の値が変わること」を分けて確認します。

配列を持つ子は、連動後の距離で個数を更新します。パラメーター自体を配列で複写する設定は扱いません。

20 挿入と分解と WBLOCK

操作	挿入基準とブロックの扱い
登録済みブロックを挿入	その定義の基点を挿入点に合わせる
ファイルからブロック挿入	図面に保存された挿入基点を使う。未設定なら原点
CAD 図形挿入	図面の原点を基準に配置し、元のブロック参照を保持
ブロック書き出し WBLOCK	座標を維持し、指定点を図面の挿入基点として保存

分解して挿入

ブロック挿入の［回転角］の下にある［分解して挿入］を選びます。初回は［しない］で、以降は前回の選択を通常終了時に保存します。

ブロック参照が入った図面ファイルを［する］で挿入すると、図面を包む外側の一階層を除きます。内部のダイナミックブロック参照は残るため、そのグリップを直接操作できます。

登録済みのダイナミックブロック自体を［する］で挿入すると、その現在形状を線や円などに分解します。入れ子のブロック参照は残ります。

配置後の分解

複数の要素を囲んで［分解］を実行すると、分解可能な要素を処理します。ダイナミックブロックも現在の形状・可視状態で一階層を分解し、動的なまとまりは失われます。

基点を持つ図面の受け渡し

JDC は図面の挿入基点を保持し、DWG・DXF では INSBASE と相互変換します。図面の挿入基点と、内部の各ブロック定義の基点は別です。同名の定義が挿入先にある場合、取り込む側に連番名を付けて既存定義を保護します。

21 DWG と DXF へ動作付きで保存する

JDesignCAD 対応範囲のダイナミックブロックを AutoCAD 2018 以降の DWG・DXF へ保存できます。両形式で同じ範囲を扱います。出力には STD 以上のライセンスが必要です。

1. 編集用の元データを JDC 形式で保存します。
2. ［名前を付けて保存］などで DWG または DXF を選びます。
3. 出力設定で AutoCAD 2018 以降の形式を選びます。
4. 保存後の案内を確認します。現在形状で保存したブロックがある場合は、名前と理由が表示されます。
5. 相手側 CAD で、形状だけでなく距離・角度・状態の変更と新規挿入も確認します。

動作付き保存の主な対象	内容
距離と回転	対応する移動・ストレッチ・回転とグリップ
尺度と配列	均等尺度、対応する距離・XY・極方向の配列
可視性と反転	可視状態、反転前後の状態
極ストレッチ	対応する伸縮枠と対象図形
ルックアップ	対応する数値表による距離・角度・XY の選択
基点とパラメーター連動	対応する基点追従、他パラメーターの移動・回転・尺度変更・端点ストレッチ、チェーン

出力処理に AutoCAD のインストールは不要です。古い版で静的な図形として書き出した DWG・DXF から、失われた動作は復元できません。元の JDC を開いて保存し直してください。

22 現在の形状で保存される設定

保存が完了しても、未対応の設定を持つブロックは AutoCAD 側でパラメーター編集ができない場合があります。警告は保存失敗とは異なり、そのブロックを現在の形状で保存したことを示します。

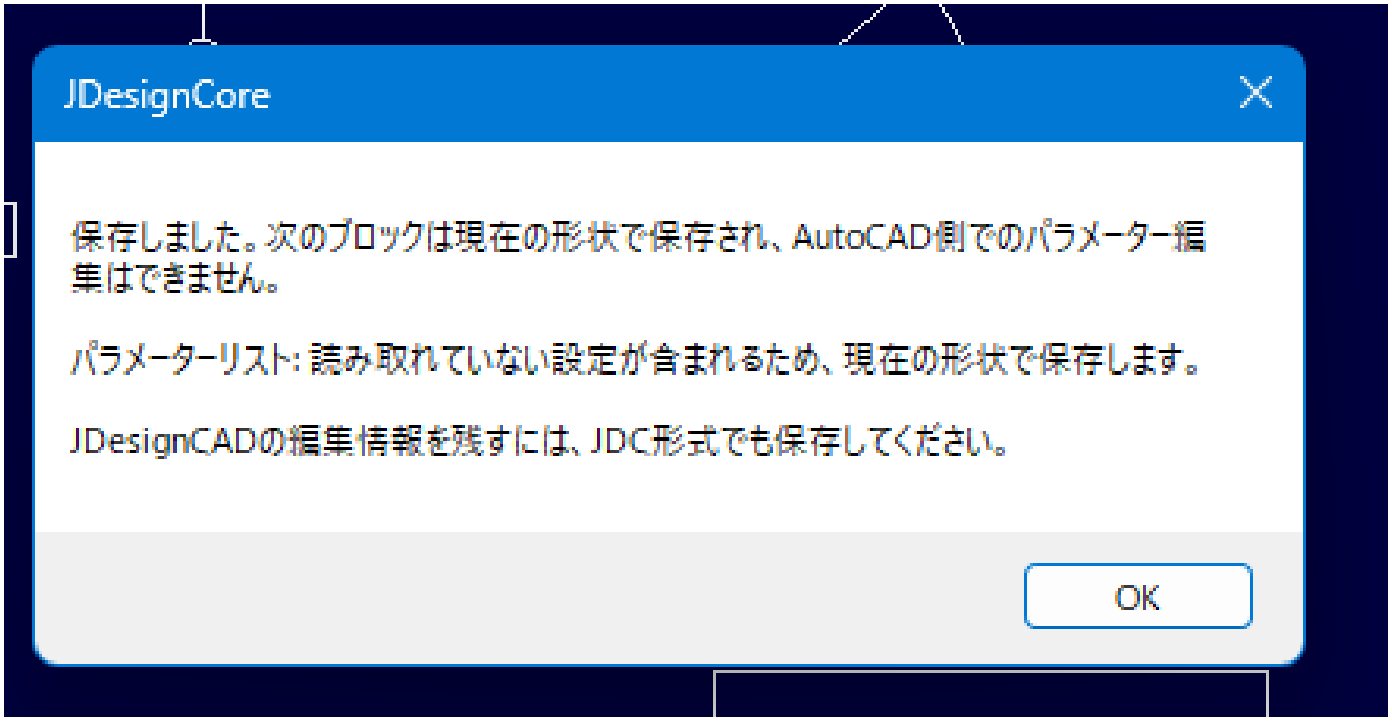


図 3-19 読み取れない設定を含むブロックの保存案内。対象名と理由を確認し、JDC の編集用データも残します。

主に次の設定が動作付き保存の対象外です。

・ブロックプロパティテーブル、幾何拘束・寸法拘束、読み取れない設定。

・位置合わせ、独立した移動アクション、極ストレッチの回転専用選択セット。

・可視性でパラメーター自体を隠す設定、非均等尺度、負の XY 距離、行または列間隔が 0 の XY 配列。

・同じ図形に独立した回転と別パラメーターの配列を設定するなど、評価結果が一致しない組み合わせ。

すべての AutoCAD 定義や動作の組み合わせとの完全互換ではありません。相手側で編集する必要がある部品は、受け渡し前に目的の操作を確認してください。JDesignCAD で編集を続ける原本は JDC で保持します。

23 FREE で使う機械部品のダイナミックブロック

test22.jdc のアイボルト形状を例に、同じブロックの長さと向きを変更します。FREE でも、図形を描き直さずに対応するパラメーターを操作し、JDC に保存できます。

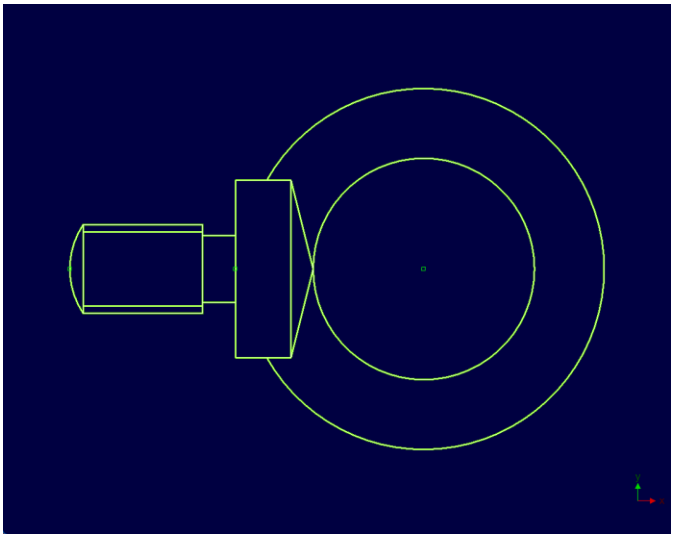


図 3-20 test22.jdc の初期状態。距離 12、回転 0 度の部品形状。

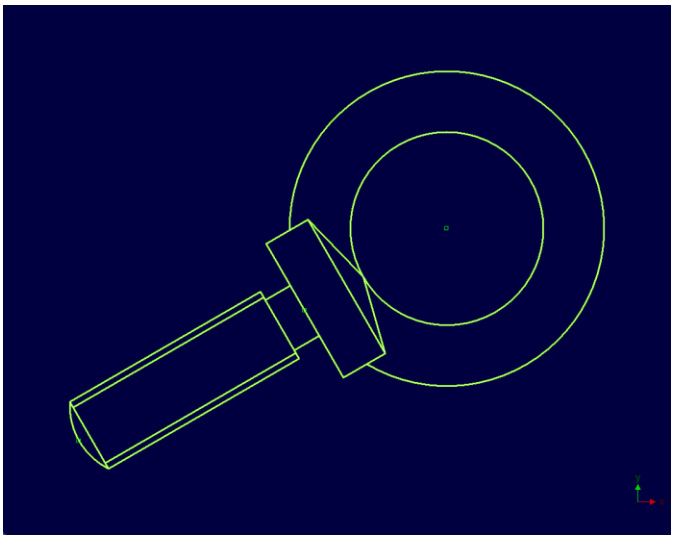


図 3-21 距離 24、回転 30 度に変更した例。リングと軸部を含む図形全体が回転しています。

1. test22.jdc を開き、部品のブロック参照を選択します。
2. 右側の［ダイナミックブロック］で［距離］を 12 から 24 へ変更します。
3. ［回転（度）］を 30 にし、形状とグリップの向きを確認します。

変更例は操作説明用です。寸法が変わった部品の規格適合性や強度を保証するものではありません。原本を残し、別名の JDC へ保存してください。

24 3D ダイナミックブロックの設計

3D の部品は、ソリッド全体を動かす動作と、ソリッドの面を変更する動作を分けて考えます。棒の長さだけを変えるなら端面の移動、部品を置き直すなら全体の移動を使います。

目的	パラメーター	動作・設定
軸・板の長さを変える	距離	3D 面の移動
壁厚・面の張り出しを変える	距離	3D 面のオフセット
面を傾ける	回転	3D 面の回転
穴・突起をもう一つ作る	距離	3D 面コピー
丸穴の径を変える	距離	3D 穴径
規格寸法を選ぶ	ルックアップ	寸法列と必要な数式を登録

利用できるエディション

3D 面動作の作成・編集には STD・Pro が必要です。Free でも、完成済みのブロックを挿入し、対応する寸法・ルックアップ・グリップを操作し、複写・JDC 保存できます。従来の 2D ダイナミックブロックの作成・編集も Free で利用できます。

作成前に決めること

1. 基準となる初期形状と初期寸法を決めます。
2. 固定したい面、動かしたい面、部品全体の向きを分けます。
3. 最小値・最大値・候補値を決め、成立しない寸法を入力させないようにします。
4. 一つの動作から始め、［動作テスト］で初期値・小さい値・大きい値を確認してから組み合わせます。

3D 面動作は登録順に初期形状から評価され、その後で部品全体の移動・回転などを適用します。寸法変更を繰り返しても、増減を累積して適用する方式ではありません。

25 実習：端面を動かして長さを変える

例として、ブロック内の X 方向に長さ 100 mm、幅 20 mm、高さ 10 mm の直方体を用意します。X＝0 側を固定し、X＝100 側の端面だけを動かして長さを変えます。以下は mm 基準の部品の例です。

1. ソリッドを選び、［ダイナミックブロックを作成…］を開きます。挿入基点を固定側に決めます。

2. 視点と作業平面を確認し、［＋距離］で X 方向の基準 2 点を指定します。表示名を「長さ」、初期距離を 100 とし、例では候補値 80;100;150 を設定します。

3. 一覧で「長さ」を選び、［動作を追加］から［3D 面の移動］を選びます。移動方向をブロック座標の＋X、倍率を 1 にします。

4. X＝100 側の端面を選び、選択数を確認して Enter または右クリックで確定します。

5. ［動作テスト］で 80、100、150 を順に指定します。固定側が動かず、反対側だけが移動することを確認します。

6. ［保存して終了］後、元の図面を JDC へ保存します。

パラメーター値	初期値との差	端面の移動	完成長さ
80	－20	－X へ 20 mm	80 mm
100	0	移動なし	100 mm
150	＋50	＋X へ 50 mm	150 mm

思った方向へ動かないとき

距離の値そのものを毎回移動量にする動作ではなく、「現在の値－初期値」が移動量の基準です。方向ベクトル、倍率の符号、選択面を確認します。値を 150 にして 150 mm 移動する設定ではありません。

面が消失するほど縮めた場合など、閉じたソリッドを保てない寸法は採用されません。直前の形状と値を確認し、成立する候補へ戻します。

26 面のオフセット・回転を登録する

オフセット：面の外向きを基準にする

距離パラメーターに［3D 面のオフセット］を登録し、対象面を選んで確定します。初期値からの増減を、各面の外向き法線へ適用します。平行な面でも外向きが逆なら、同じ正の増分で逆方向に動きます。

動作	方向の基準	用途
面の移動	設定した共通ベクトル	複数の端面を同じ方向へ動かす
面のオフセット	それぞれの外向き法線	表面を外へ広げる、内へ狭める

穴の内壁では、ソリッドから見た外向きは穴の空間側です。円筒穴の寸法変更には［3D 穴径］を使うと、パラメーター値を直径として扱えます。

回転：軸と初期角度を決める

1. ［＋回転］で中心と基準方向を指定し、初期角度・候補値を確認します。
2. ［動作を追加］から［3D 面の回転］を選びます。
3. 回転軸と倍率を設定し、対象面を選択して確定します。軸はブロック内の 3D 座標を基準にします。
4. ［動作テスト］で正負の小さい角度を指定し、固定したい箇所が保たれることを確認します。

成立範囲と動作の順序

面回転は部品全体の剛体回転とは異なり、周囲の面との接続を作り直します。大きい角度、薄肉、穴やフィレットとの交差では成立しないことがあります。必要な寸法の組み合わせをテストし、形状が成立する範囲で制限値を設定します。

前の動作で後の対象面が消える設定は避けます。多数の動作を一度に登録せず、一つずつ保存前のテストで確認してください。

27 穴径変更と穴・突起のコピー

穴径の動作

距離パラメーターの初期値を元の穴径に合わせ、［3D 穴径］を関連付けます。穴要素または対応する円筒内面を選び、テストで直径を変更します。この動作は距離値そのものを直径に使います。

同じ穴コマンドで複数位置を作った穴では、その組の穴を一緒に変更します。一つだけ変更する設計なら、作成時の穴グループも分けておきます。ねじ規格やねじ山径まで自動で変更する機能ではありません。

面コピーの動作

1. 距離パラメーターを追加し、初期値を決めます。

2. ［3D 面コピー］を追加し、移動方向と倍率を設定します。

3. 穴、または突起を構成する側面・端面を選び、確定します。

4. ［動作テスト］で初期値より大きい値を入力し、元形状を残して一つコピーされることを確認します。

初期距離	入力距離	コピーの位置
10 mm	10 mm	初期状態。コピーを追加しない
10 mm	30 mm	元の穴・突起から指定方向へ 20 mm
10 mm	40 mm	初期形状から 30 mm の位置へ一つ

値を順に変えてもコピーが増殖するわけではありません。等間隔に多数を並べる配列とは用途が異なります。

選択と形状の条件

突起は必要な構成面をまとめて選び、コピー先でも本体につながる形状にします。任意の面を独立したサーフェスへ複写する操作ではありません。穴編集情報を持つソリッドでは、面コピーの対象は穴です。読み込み形状などで認識できない場合は、通常の穴作成や別のモデル構成を検討します。

28 複数ソリッドの面を一つの動作で動かす

同じブロック内の複数ソリッドから選んだ面を、一つの 3D 面動作へ登録できます。ソリッドは結合されず、色・画層・部品属性を保ちます。

実習：二つの部材の境界を移動する

X 方向の 0〜40 mm を部材 A、40〜100 mm を部材 B とし、X＝40 の平面で接する二つの直方体を作ります。両端を固定したまま、境界だけを 50 mm へ動かす例です。

1. 二つのソリッドを同じブロックに含め、ブロックエディターを開きます。

2. 初期距離 40 の「境界位置」パラメーターを用意し、［3D 面の移動］を追加します。

3. 方向を＋X にし、A の右端面と B の左端面を選びます。重なった境界を窓選択し、対象が「2 面」であることを確認します。

4. Enter で登録し、動作テストで境界位置を 50 にします。

5. A は長さ 50、B は長さ 50 になり、全長 100 と両端の位置が変わらないことを確認します。

確認対象	期待する状態
接合位置	両方の端面が同じ X 座標にある
全長	100 mm のまま
構成	二つのソリッドを維持
色・属性	A と B の設定を維持

操作上の注意

面の移動・オフセット・回転・コピー・穴径で共通の対象登録を使いますが、各ソリッドがそれぞれの動作条件を満たす必要があります。どれか一つが失敗した場合は、全対象の形状と入力値を元の状態に保ちます。

入れ子ブロック内のソリッドは、その内側のブロックを開いて設定します。見た目が一体でも、対象が別の定義内にないか確認してください。

29 ルックアップ表で数式を使う

ルックアップは、規格名から寸法の組を選ぶ表です。同じ行の別の列を参照して値を計算できます。数式は JDC へ保存されます。

1. 先に「長さ」「Lg」などのパラメーターと動作を登録します。

2. ［＋ルックアップ］で表示名とグリップ位置を指定し、表の入力項目を選んで［入力項目を反映］します。

3. 各行に名前、長さ、Lg の数式を入力します。セル上にマウスを置いて計算結果を確認できます。

4. ［確定］後、［動作テスト］で名前を選び、複数の寸法が同時に変わることを確認します。

名前	長さ	Lg に入力する内容	Lg の結果
規格 20	20	長さ*0.8	16
規格 30	30	(長さ-5)/2	12.5
規格 40	40	=長さ*0.8	32

入力できる式

四則演算の＋、－、*、/、括弧、正負の符号を使えます。×、÷、先頭の＝も使用できます。空白や記号を含む列名は［回転（度)］のように角括弧で囲みます。実際の入力では半角の［ と ］を使い、列見出しと同じ名前にします。

右側の列も参照できますが、別行の値は参照しません。長さ→Lg→余白という参照は可能です。長さ→Lg→長さという循環、ゼロ除算、不明な列名、値の範囲違反は確定できません。

常時連動する数式との違い

数式は「表の行を計算する」機能です。表にない任意寸法へ直接変更したときに、常にすべての関連式を解く機能ではありません。規格の選択で寸法をそろえる運用に使い、自由入力と混在させる場合は次項の名称判定を確認します。

30 規格名の判定と表のメンテナンス

数値を直接変更した場合も、登録した行に一致すれば規格名を表示し、一致しなければ「カスタム」などの名称を表示します。規格名と実際の寸法が一致しているかをテストしてください。

判定に使う列を選ぶ

［名称の判定に使う］は、どの寸法で行の名前を判定するかを指定します。例えば長さで規格が決まり、Lg は付随寸法なら、Lg を判定対象から外す運用ができます。ただし、異なる行を区別できない判定条件は避けます。

設定・操作	働き
名前を選ぶ	行に対応する寸法・角度を一括適用
数値を直接変える	一致する行の名前、またはカスタムを表示
読み取り専用	数値から名前を表示。名前による寸法変更は不可
入力項目を変更	同じ項目の値を保持。追加列の空欄は入力が必要
行見出しを選び Delete	選択した行を削除
位置を指定し直す	ルックアップのグリップ位置を変更

表を変更した後の配置済み部品

ルックアップ表を変更しても、配置済みの有効な寸法・角度を勝手に新しい規格値へ変えません。表に一致しなくなった配置はカスタムになります。新しい規格に切り替える場合は、配置のプロパティで名前を選び直します。

候補値の削除などで保持できない値がある場合は、保存前の案内を確認してください。定義を上書きすると、その定義を使う配置へ反映します。別製品として分けたい場合は［別名で保存］を使用します。

よくある入力エラー

名前の重複、曖昧な寸法の組、未入力のセル、最小・最大・候補値に反する結果を確認します。式の列名は見出しをそのまま使います。範囲指定やワイルドカードなど、任意の AutoCAD ルックアップを同じように編集できるわけではありません。

31 可視状態とグリップ表示をそろえる

平面図・側面図や、閉・半開・開などの状態を、同じブロックで切り替えます。可視状態は図形の表示、［パラメーターの表示］は状態ごとのグリップ表示を設定します。

図形の状態を登録する

1. ブロック内に必要な図形を作り、［＋可視性］を追加します。可視性パラメーターは一つです。

2. 表示したい図形を選び、［状態を追加］で「平面」などの名前を付けます。

3. 別の図形を選び「側面」などを追加します。既存状態を直す場合は［選択を状態に登録］を使います。

4. ［初期状態にする］で挿入時の状態を選びます。図形を選ばず登録すると、その状態の管理対象図形はすべて非表示になります。

状態に不要なグリップを隠す

1. ［動作の設定］で可視性パラメーターと可視状態を選びます。

2. ［パラメーターの表示］を開き、表示したい項目だけをチェックします。

3. OK で登録し、他の可視状態にも必要な設定を行います。

4. ［動作テスト］と通常の配置で切り替え、図形とグリップの両方を確認します。

平面で見せる例	側面で見せる例
幅、奥行、平面用ルックアップ	高さ、側面用ルックアップ
可視状態を切り替えるグリップ	可視状態を切り替えるグリップ

可視性の切替グリップ自体は隠せません。全体の設定で非表示にしたグリップは、個々の状態でも表示しません。

非表示になるのは操作用グリップです。プロパティの値、ルックアップ、動作計算、挿入基点は維持します。旧ブロックは従来のグリップ表示を引き継ぐので、不要な項目をこの画面で設定してください。

32 保存形式と受け渡しの判断

保存先・用途	保持できるもの	運用
JDC	3D 面動作、数式、可視状態、計算済み形状	再編集する原本
対応する 2D 動作の DWG／DXF	対応範囲の動的定義・配置	相手側で必要な動作を確認
ルックアップ数式を DWG／DXF へ	計算済みの数値	数式自体の再編集は JDC で行う
3D 面動作を含むブロックの DWG／DXF	現在の形状	同じ定義内の動作情報は静的化
IFC ライブラリ書き出し	現在の可視状態・形状と部品属性	詳細は第 5 部。動作定義は IFC へ渡さない

分解が必要な場合と不要な場合

部品として繰り返し使う原本は分解せず JDC に残します。IFC へ渡すときは、ソリッド群を含む対応ブロックをそのまま選べます。現在の可視状態のソリッド群を一つの部品の形状として出力します。

通常の分解を行うと、配置の形状を個別図形へ取り出します。元のダイナミック定義による操作を続けたい場合は、分解前のブロックを別名保存しておきます。

DWG／DXF 読み込み後の編集

他ソフトで変更済みの匿名配置は、評価済みの外観を保って読み込む場合があります。［初期状態に戻してダイナミック編集を開始］を使うと初期状態へ戻ります。現在の形状を保ったまま任意の元定義を再編集できるという意味ではありません。

未対応の拘束・動作・組み合わせがある場合は、現在の形状で保存されるか、編集対象外となります。JDC を原本、交換形式を受け渡し用とし、相手側で動作を必要とするか、外観だけでよいかを決めて出力します。

33 ダイナミックブロックの確認表

症状	先に確認すること	次の操作
パラメーターだけ動き、図形が変わらない	動作との関連付け、対象図形／面	［動作を追加］または対象を設定し直す
ストレッチが終わらない	枠指定後の対象選択段階か	図形を選び Enter／右クリックで確定
円が変形しない	曲線をまたぐ部分ストレッチか	全体移動または適切な寸法・形状編集を使う
3D が逆方向へ伸びる	初期値、方向、倍率の符号	小さい増減で動作テスト
一部寸法で 3D 編集が失敗する	薄肉・交差・対象面の消失	成立範囲を確認し候補値を制限
二つのソリッドの片方しか動かない	対象が 2 面か、入れ子でないか	内側の定義と選択数を確認
状態切替後も不要なグリップが残る	可視状態ごとの表示設定	［パラメーターの表示］で設定
表を直しても配置済み寸法が変わらない	配置の値が保持されているか	規格名を選び直す
他の配置まで変わった	配置値の変更か定義の変更か	別製品は定義を別名保存
DWG では 3D 動作がなくなる	3D 面動作を含む定義か	編集用 JDC を渡す／保持する

配布前の動作確認

初期値、最小値、最大値、全ルックアップ行、全可視状態を試します。移動・回転して配置した状態、複写、元に戻す／やり直し、JDC の保存・再読込も確認してください。

形状が成立することと、製品規格に適合することは別です。規格表にない寸法を許す場合は、部品名だけで規格品と判断されないように運用します。

第 4 部 解析機能

STD の主な機能として、CAD で作成した形状を使い、2D・3D の応力解析、定常・非定常熱解析、梁計算、トラス、ラーメンを実行できます。変位・応力・反力・ひずみや温度分布を確認し、アニメーションと資料出力に活用します。STD・Pro で利用できます。



図 4-A ［解析］タブの操作例。2D・3D の各コマンドにアイコンがあります。

01 材料を登録する 313
02 解析する形状を整える 314
03 2D 解析の基本手順 315
04 3D 解析の基本手順 316
05 面と稜線と頂点を選ぶ 318
06 2D の部分選択と局所メッシュ 319
07 固定と荷重を設定する 320
08 結果の項目と単位 321
09 変形をアニメーション表示する 322
10 変形前の輪郭を重ねる 323
11 メッシュ線と滑らかな色表示 324
12 反力を確認する 325
13 ひずみを確認する 326
14 定常熱解析を行う 327
15 非定常熱解析を行う 328
16 未設定の境界を一括で放熱にする 329
17 時刻別の結果を再生し保存する 330
18 梁計算を行う 331
19 3D 梁の座標と断面 332
20 トラスとラーメン 333
21 画像と印刷と数値を出力する 334
22 条件を図面に保存する 335
23 リブと穴のある部品の解析例 336
24 フランジと曲面を含む部品の解析例 337

対応する計算

面・ソリッドの応力解析は微小変形の線形弾性です。熱解析は定常と非定常の熱伝導に対応します。2D は 8 節点四角形、3D は 10 節点四面体のメッシュを使用します。

3D では複数材料と完全接合に対応します。非線形、塑性、座屈、振動、離間・滑りの接触、熱応力連成、自重の自動荷重化、設計基準による合否判定は含みません。

第 4 部 詳細目次：条件設定と実習

01〜24 の画面操作に加え、解析モデルの選び方、複数材料、確認用の例を説明します。

25 目的に合う解析を選ぶ 338
26 解析条件を組み立てる順序 339
27 複数材料と完全接合を設定する 340
28 メッシュ寸法と計算精度 343
29 荷重の入力を間違えない 344
30 実習：梁計算を既知の値と比較する 345
31 3D 梁の局所軸・ねじり定数 346
32 トラス・ラーメンの接続と結果 347
33 定常熱解析の境界条件を読む 348
34 実習：非定常熱解析で冷却を見る 349
35 変位・応力・反力を読み分ける 350
36 条件・結果の保存と出力 351
37 解析で困ったときの確認表 352

01 材料を登録する

［解析］→［基本設定］→［材質リスト登録］で材料を追加・編集・複製・削除できます。初期材料には SS400、S45C、SCM440、SUS304、SUS316、SUS430、A5052-H112、A6061-T651、C1100、C2801 があります。

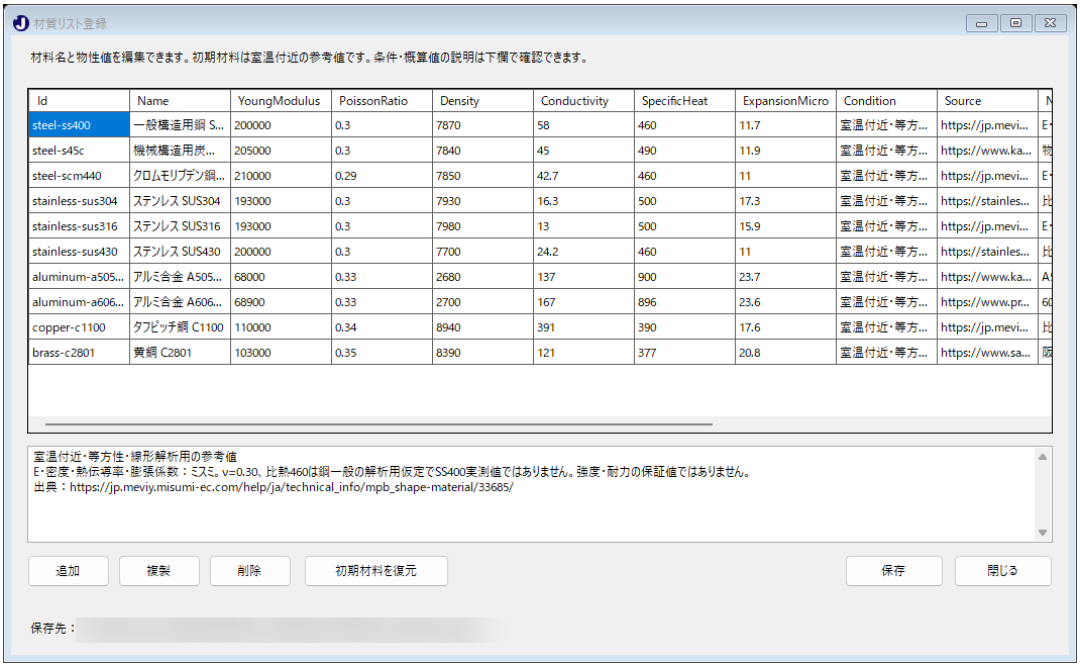


図 4-1 材質リスト登録。材料を選び、物性値と下部の説明を確認します。保存先の表示は環境により異なります。

物性	入力単位と用途
ヤング係数	N/mm²。線形弾性の剛性
ポアソン比	無次元。横方向の変形
密度	kg/m³。非定常熱解析で使用
熱伝導率	W/(m K)。熱伝導
比熱	J/(kg K)。非定常熱解析の蓄熱
線膨張係数	10^{-6}/K。登録はできるが現在の計算には使用しない

初期値は室温付近の参考値です。材料名だけで物性一式が保証されるものではありません。保存した解析には当時の材料値を保持します。共通リストの新しい値を使う場合は材料を選び直して再計算します。

02 解析する形状を整える

細かな形状が解析目的に不要な場合は、取り込み前に［図形の最適化］を使います。元の図形を変更するため、原形を残す場合は先に複写してください。

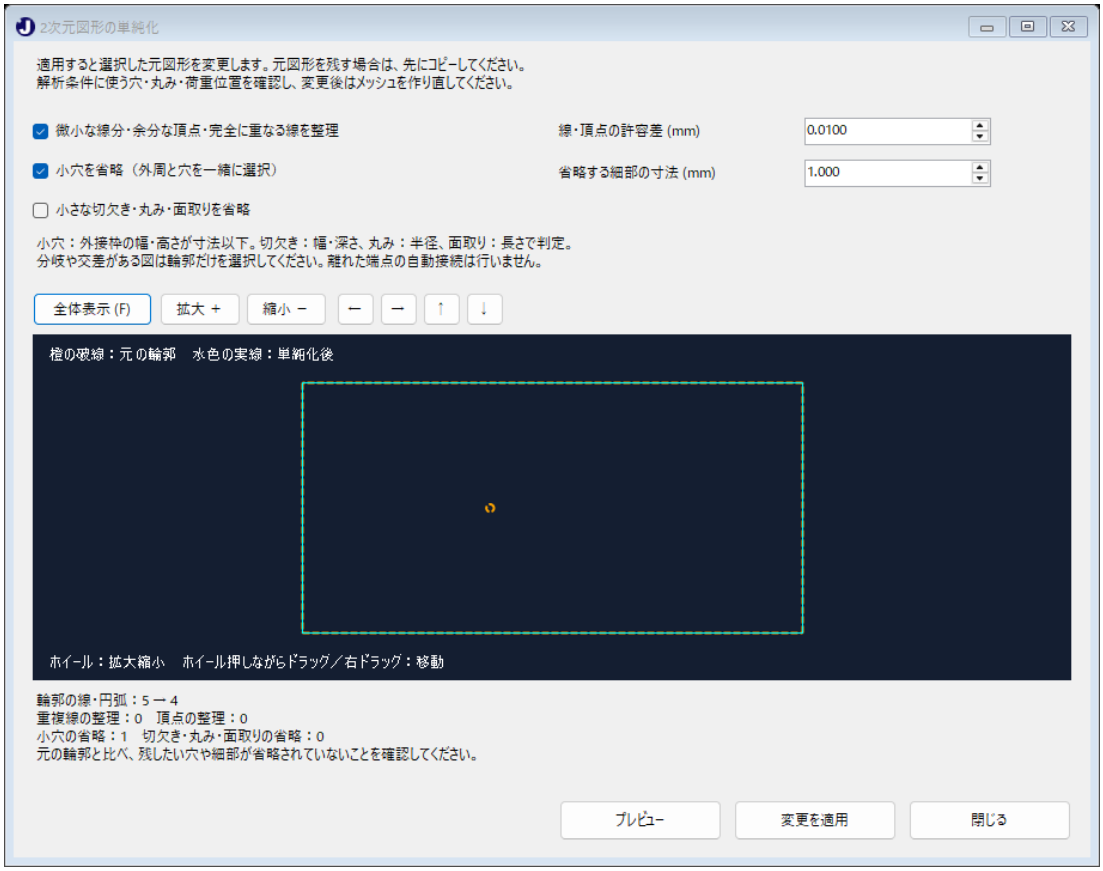


図 4-2 2D 輪郭の単純化。プレビューで元の輪郭と変更後を比較してから適用します。

1. 同じ平面上の外周と穴、または平面の面を選択します。
2. ［図形の最適化］→［2 次元の輪郭を単純化］→［2 次元の設定へ］を開きます。
3. 線・頂点の整理、重複線の整理、小穴・細部の省略から必要な処理を選びます。
4. ［プレビュー］で橙の破線と水色の実線を比較し、［変更を適用］します。

3D は既存のフィレット削除や面の削除・補完を使い、対象を選んで実行します。小穴を寸法から自動検出して一括除去する機能ではありません。

荷重や固定の位置、応力集中を評価する穴や丸みまで省略すると、解析する問題が変わります。適用後は図形を取り込み直し、メッシュと条件を設定してください。

03 2D 解析の基本手順

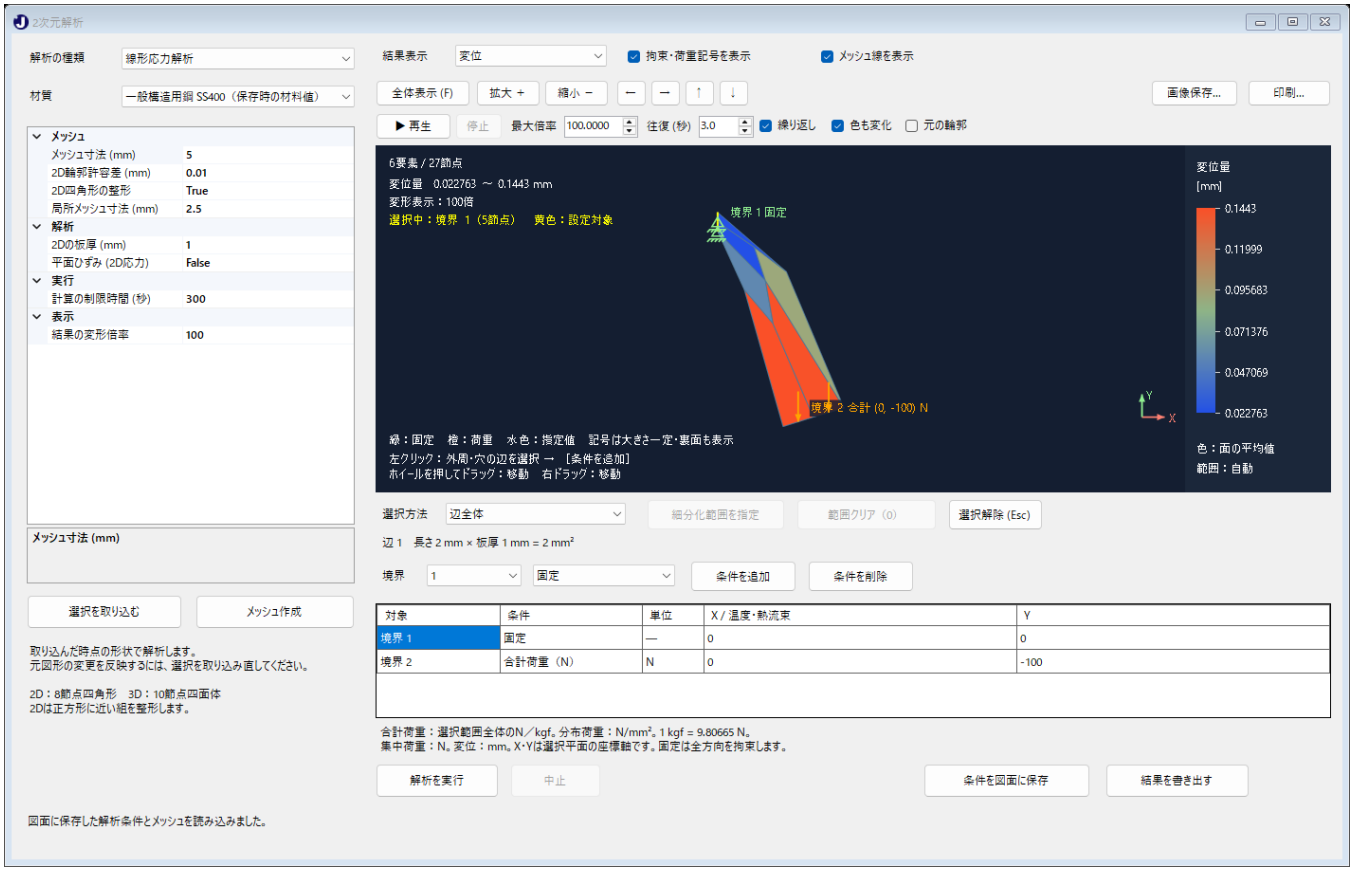


図 4-3 2 次元解析画面。左側で材料・メッシュなどを設定し、下部の条件表から解析します。

1. CAD で平面の面を一つ、または同じ平面上の外周と穴の輪郭を選択します。
2. ［解析］→［2 次元解析］のメッシュ・応力解析・熱解析から画面を開き、［選択を取り込む］を押します。
3. 解析の種類、材料、メッシュ寸法、板厚を設定して［メッシュ作成］を押します。
4. 辺や節点を選び、固定・荷重などの［条件を追加］を押します。表の数値を入力して Enter で確定します。
5. ［解析を実行］し、結果表示を切り替えます。

応力解析は平面応力または平面ひずみを選びます。2D の X・Y は取り込んだ平面内の座標で、板厚方向が Z です。外周は一つで、離れた領域や穴の中の島は別々に扱います。

CAD 側を変更しても取り込んだ形状は自動更新されません。変更後は再度［選択を取り込む］を実行してください。

04 3D 解析の基本手順

3D では閉じたソリッドを 1〜100 個選びます。［3 次元解析］→［選択を取り込む］で取り込み、材料・接合とメッシュ寸法を設定して［メッシュ作成］へ進みます。複数材料は本部 27 を参照してください。

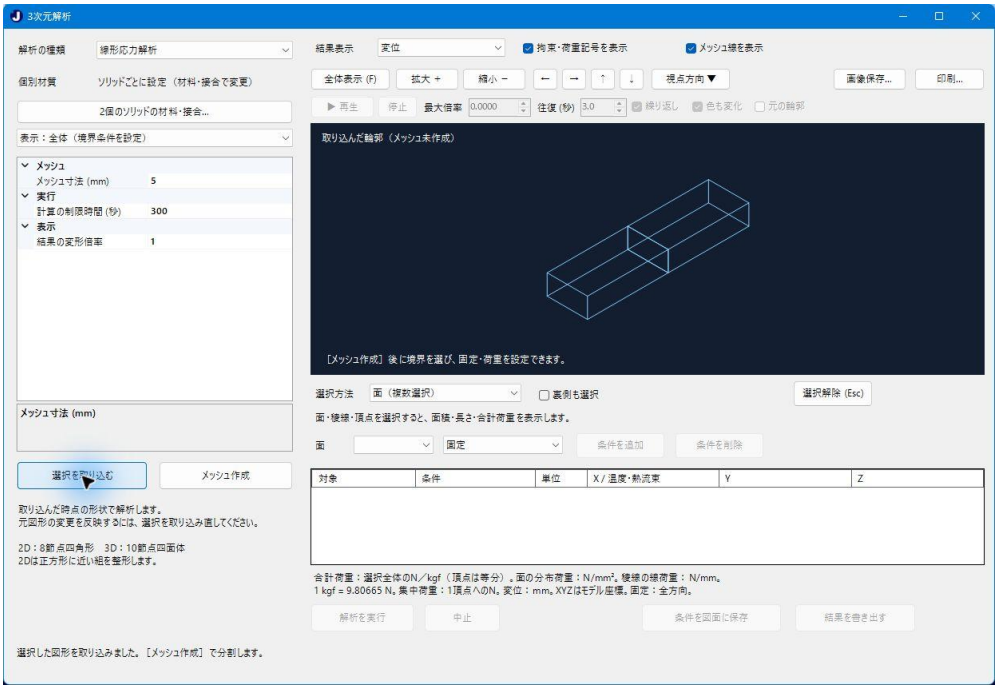


図 207　3D 解析の取込み画面。材料・接合と表示対象をここで設定します。

以下の結果図は単一部品の解析例です。現在の材料・接合ボタンとアセンブリの設定手順は、上の画面および本部 27 を参照してください。

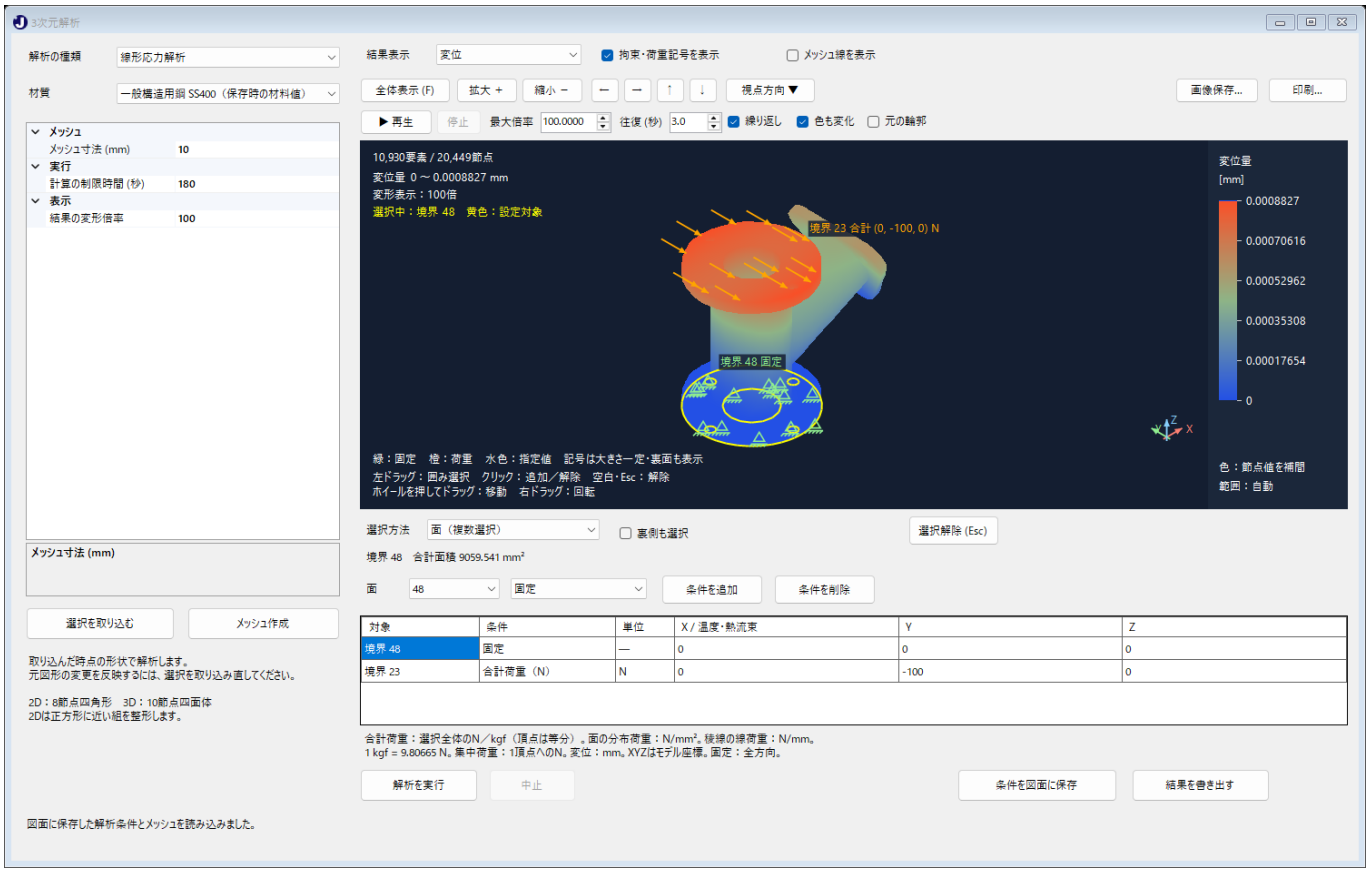


図 4-4　メッシュ作成と条件設定後の 3D 変位表示。面・稜線・頂点はメッシュ作成後に選択できます。

メッシュ作成後に面・稜線・頂点を選び、条件を追加します。3D の荷重や変位の X・Y・Z はモデル座標です。条件表で成分と単位を確認してから［解析を実行］します。

選択やドラッグが効かないとき

左上に「取り込んだ輪郭（メッシュ未作成)」と出ていないか確認します。条件を付ける対象はメッシュ作成後に選びます。CAD 側の選択と解析画面内の境界選択は別の操作です。

拘束が不足して物体全体が動いたり回ったりできる場合は計算できません。実際の取付状態に合う面や点を拘束してください。

05 面と稜線と頂点を選ぶ

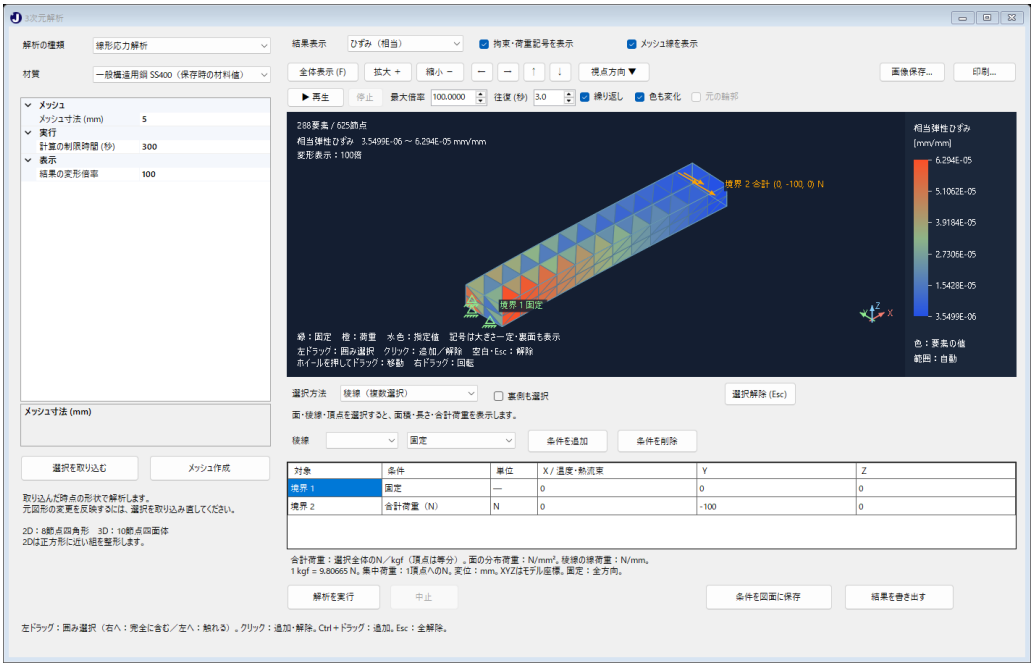


図 4-5 ［選択方法］を稜線に切り替えた画面。選んだ稜線は黄色で表示します。

操作	選択する対象
クリック	追加。選択済みなら解除
左から右へドラッグ	枠に完全に入る対象
右から左へドラッグ	枠に触れる対象も含む
Ctrl＋ドラッグ	現在の選択を残して追加
裏側も選択	囲み選択で裏側の対象も含む
空白クリック　または　Esc	選択表示だけを解除。条件と結果は保持

［選択方法］で面・稜線・頂点を切り替え、同じ種類を複数選べます。稜線は CAD 形状の辺で、表面の三角メッシュの対角線ではありません。面は面積、稜線は長さ、頂点は点数を表示します。

頂点の番号は対応するメッシュ節点番号です。3D の任意の内部節点や、面の一部分を直接選択する機能とは異なります。

06 2D の部分選択と局所メッシュ

2D の［選択方法］では［辺全体］［辺の範囲（2 点）］［節点］を使い分けます。集中荷重では節点を選びます。

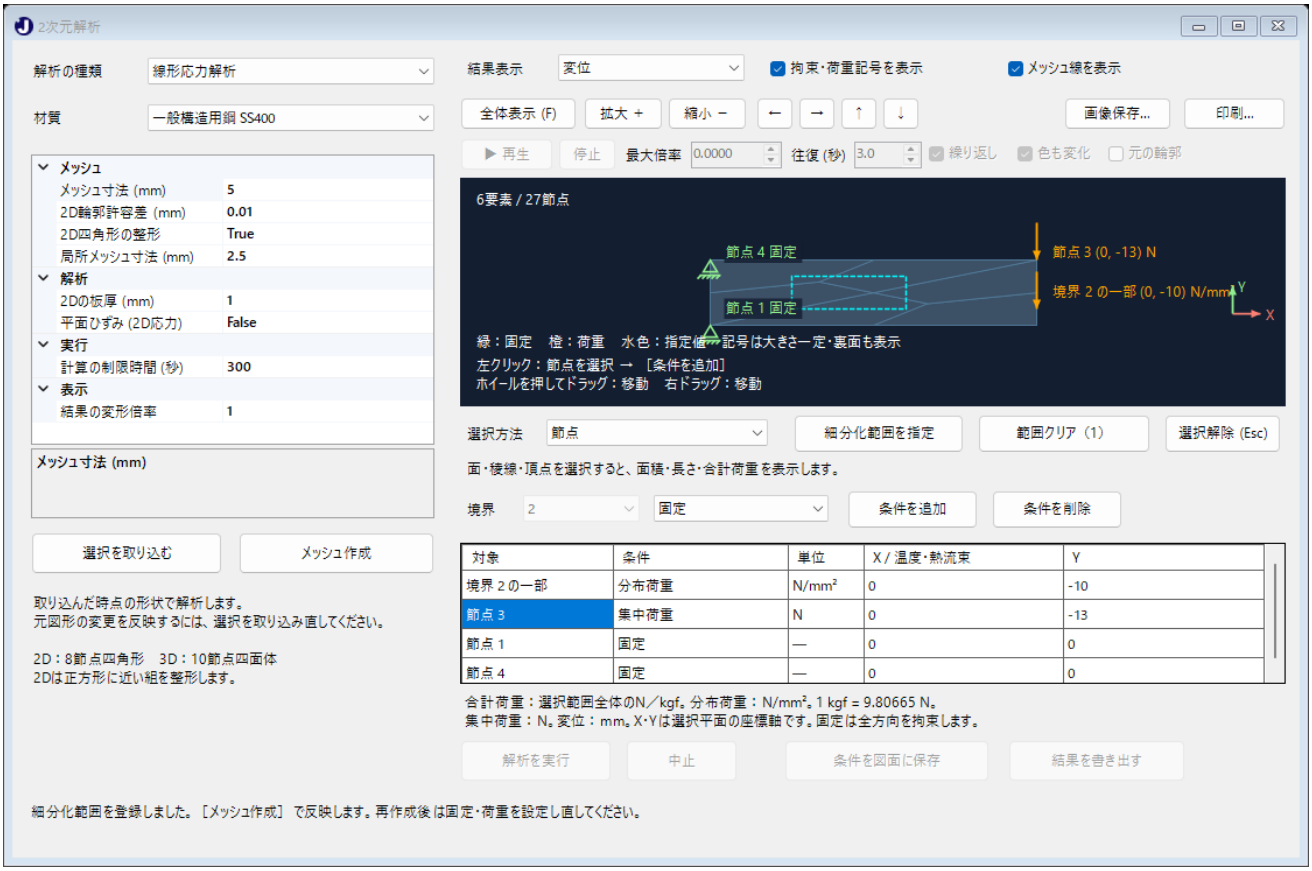


図 4-6 局所的に細かくする範囲の指定例。範囲を指定した後にメッシュを作り直します。

辺の一部分を選ぶ

［辺の範囲（2 点）］で同じ境界辺上の 2 点をクリックします。選択区間はメッシュの区切りに沿って決まるため、より細かい範囲が必要ならメッシュを細かくします。閉じた辺では短い側を選びます。

局所的にメッシュを細かくする

［局所メッシュ寸法］を設定し、［細分化範囲を指定］から矩形で範囲を指定します。最大 32 範囲を登録でき、［メッシュ作成］で反映します。［範囲クリア］は次回作成時に反映します。

メッシュを作り直すと節点・境界区間の番号が変わるため、固定や荷重を再設定します。3D の局所細分化は対象外です。

07 固定と荷重を設定する

対象	応力解析の主な条件
2D の辺・区間	固定、変位指定、合計荷重、分布荷重
2D の節点	固定、変位指定、集中荷重
3D の面	固定、変位指定、合計荷重、分布荷重
3D の稜線	固定、変位指定、合計荷重、線荷重
3D の頂点	固定、変位指定、合計荷重、集中荷重

対象を選択して条件を追加し、表で X・Y・Z の数値を入力します。固定は全並進成分を 0、変位指定は全並進成分を指定します。連続体の軸ごとの自由・拘束の切替とは異なります。

合計荷重と分布荷重

合計荷重は選択した範囲全体の力です。2 面に 100 N を指定しても、各面へ 100 N ずつ掛かる設定ではありません。面は面積、稜線は長さ、複数頂点は点数に応じて配分します。

分布荷重は N/mm²、線荷重は N/mm、集中荷重は 1 点への N です。合計荷重は N または kgf を選べます。1 kgf＝9.80665 N です。2D の荷重面積は辺の長さ×板厚です。

入力例

負の Y 方向へ合計 100 N なら、合計荷重を選び X＝0、Y＝－100（3D は Z＝0）とします。これは一定方向の力で、曲面の各位置で自動的に法線方向へ向く圧力ではありません。

同じ場所の荷重は加算します。条件の種類を置き換える場合は古い行を削除してください。荷重矢印は見やすい一定長で、長さが力の大きさを表すわけではありません。

08 結果の項目と単位

計算後は［結果表示］から項目を切り替えます。材料・形状・境界条件を変えない表示切替では再計算しません。

結果	意味と単位
変位	変位ベクトルの大きさ。mm
応力 ミーゼス	相当応力。N/mm² = MPa
反力	固定・指定変位の拘束が物体に加える力。N
ひずみ 相当	相当弾性ひずみ。mm/mm
ひずみ X Y Z	方向別の伸び縮み。mm/mm
せん断ひずみ XY YZ XZ	工学せん断ひずみ。2D は XY
温度	°C
熱流束	W/m²

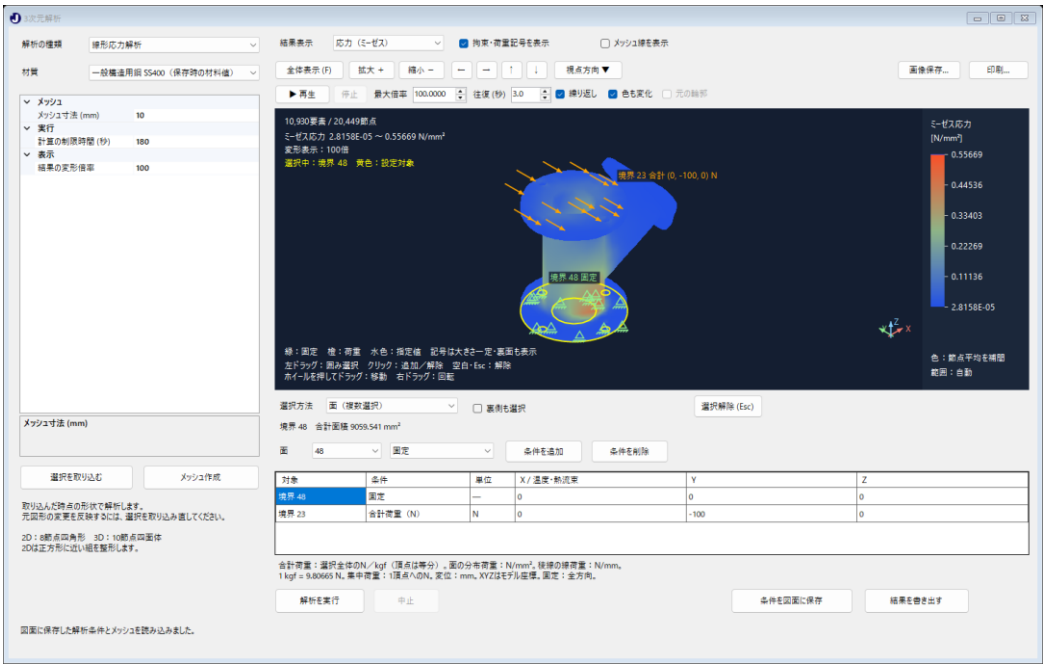


図 4-7 3D のミーゼス応力表示例。凡例の単位と色範囲を確認します。

変形倍率は形の見せ方に掛ける倍率です。変位・応力・ひずみ・反力などの計算値には掛けません。凡例の範囲は表示対象から決まり、内部を含む全要素の極値とは一致しない場合があります。

09 変形をアニメーション表示する

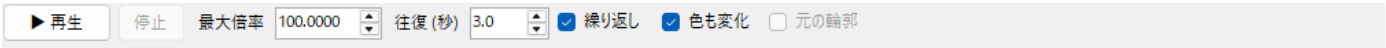


図 4-8 2D 変形アニメーションの操作欄。形状と色の変化を表示設定で調整します。

1. 線形応力解析を実行し、［変位］［応力（ミーゼス）］またはひずみを表示します。
2. ［最大倍率］を設定します。1 倍は計算された変位で、数値を大きくすると形状変化を強調します。
3. ［往復（秒）］で 1 往復の表示時間、［繰り返し］で連続再生を設定します。
4. ［再生］を押します。元の形→変形後→元の形の順に表示します。

［色も変化］をオンにすると、形状と同じ割合で 0 の色から結果の色へ変化します。オフでは結果の色を維持します。［一時停止］はその時点を保持し、［停止］は最大倍率の静止結果へ戻します。

これは解いた結果を補間する表示であり、荷重の時間変化や振動を計算したものではありません。反力は静止表示です。大きなモデルで表示が重い場合はメッシュ線をオフにします。動画ファイルへの保存は対象外です。

10 変形前の輪郭を重ねる

［結果表示］を［変位］にして［元の輪郭］をオンにすると、変形前の輪郭を破線で重ねます。資料で元の位置と変形量を比較するときに使います。

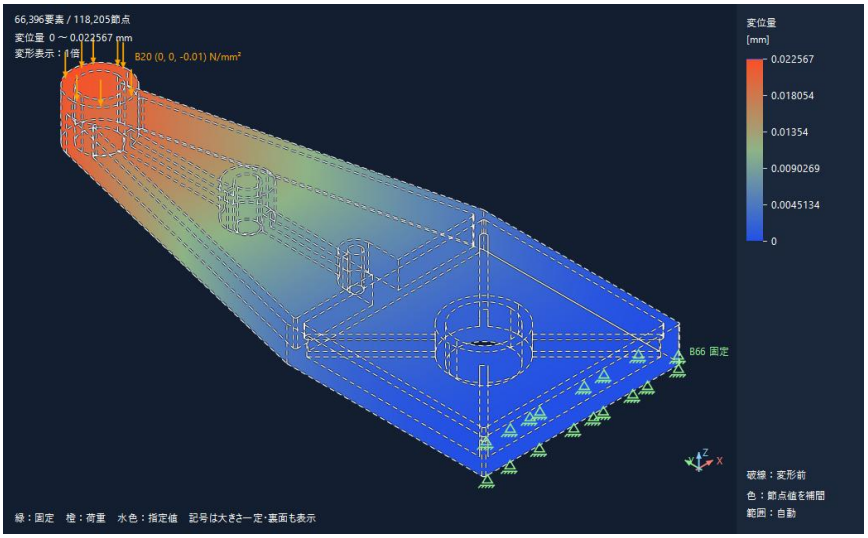


図 4-9 変形前の破線と変形後の形状を重ねた表示例。

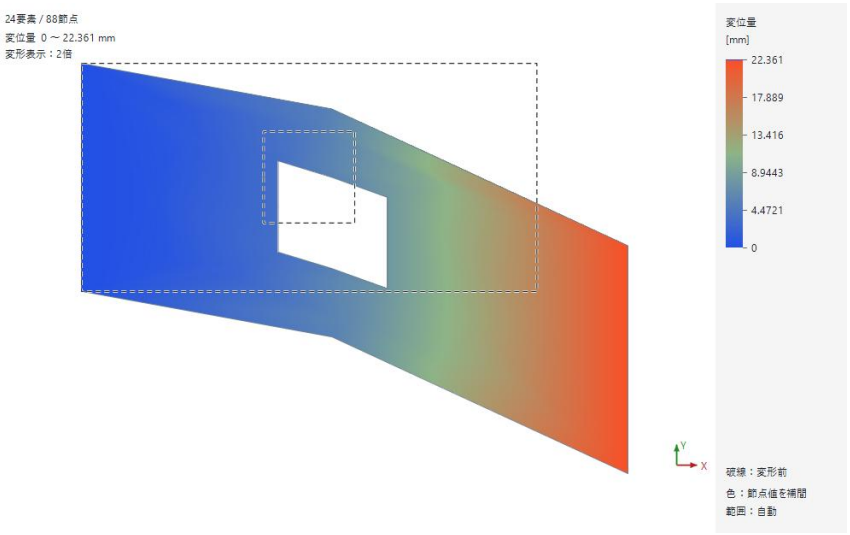


図 4-10 白背景で出力した 2D の輪郭比較例。印刷でも破線を表示します。

2D は外周と穴、3D は面の境界と視点に応じた外形を表示します。基準は解析メッシュの変形前形状です。3D は裏側の線も重ねて比較します。

［全体表示］で両方の形を収めてください。倍率やアニメーションで変形前の線は動きません。画像保存・印刷にも反映します。応力・反力・ひずみや熱解析ではこの重ね表示を使いません。

11 メッシュ線と滑らかな色表示

［メッシュ線を表示］をオフにすると、網目を消し、節点間で色を補間した滑らかな分布を表示します。



図 4-11 要素ごとの色で分布を確認する表示例。

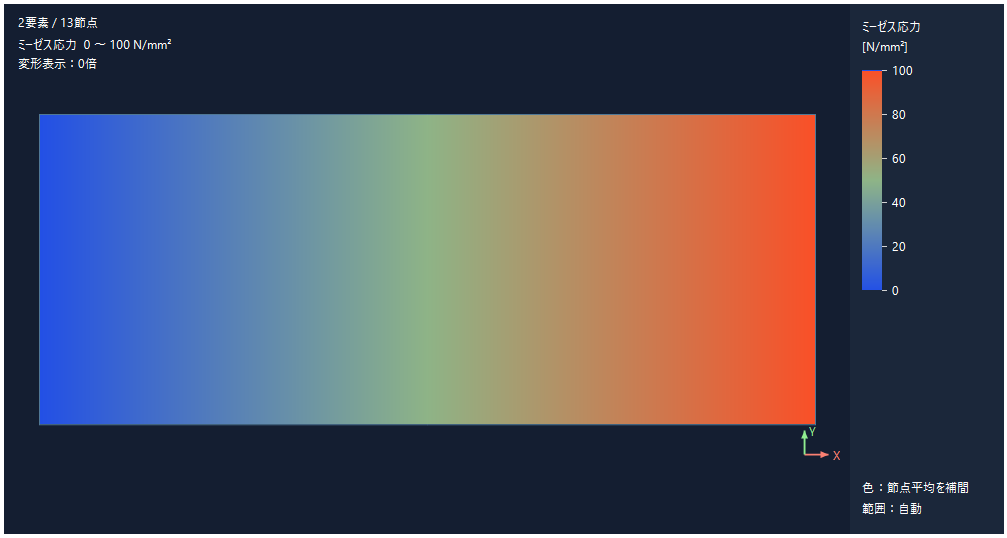


図 4-12 同じ結果を滑らかな色で表示した例。表示方法の違いで計算値は変わりません。

変位・温度は節点値を補間します。応力・ひずみ・熱流束は、同じ節点につながる要素の表示値を平均して補間します。凡例に表示方法を併記します。

固定・荷重記号と黄色の選択表示は別の設定なので残ります。結果だけを見たい場合は［選択解除］を押し、必要に応じて［拘束・荷重記号を表示］もオフにします。

12 反力を確認する

［結果表示］→［反力］で、拘束が物体に加える力を紫の矢印で示します。右側には全拘束の合計 Rx・Ry・Rz を N で表示します。2D は Rx・Ry です。

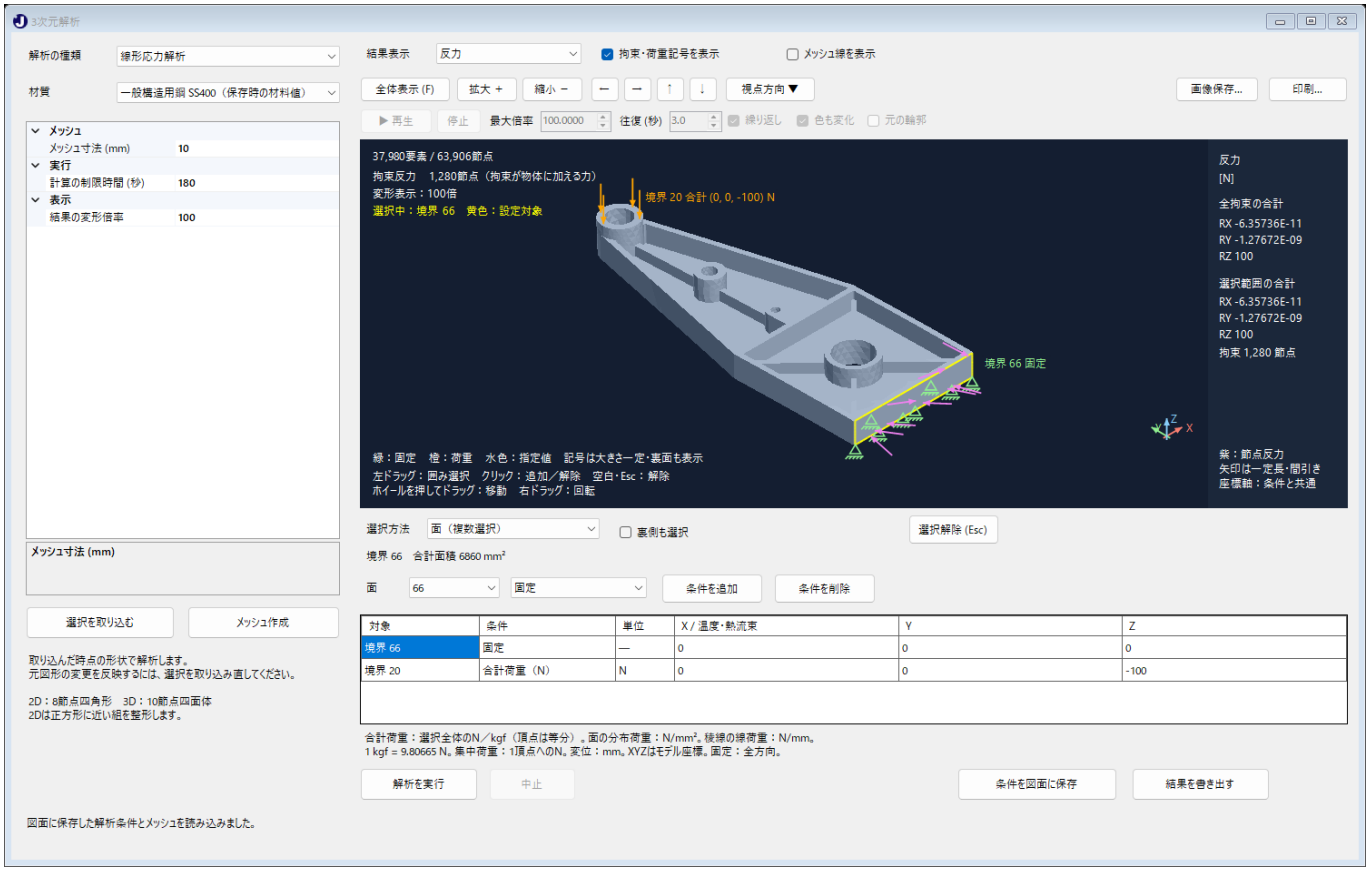


図 4-13 3D 反力の表示例。選択範囲の合計と全拘束の合計を区別します。

面・辺・節点などを選ぶと、その範囲に含まれる拘束節点の反力合計も表示します。拘束のない範囲は 0 です。方向は条件入力と共通で、2D は解析平面、3D はモデル座標です。

矢印は向きを示す一定長で、密な箇所は間引いて表示します。合計値や CSV は間引かず、共有節点を重複して加算しません。

［結果を書き出す］のファイル種類で［節点反力 CSV］を選ぶと、変形前の座標、方向別反力、反力の大きさを保存します。画像保存・印刷にも反力表示を反映します。

13 ひずみを確認する

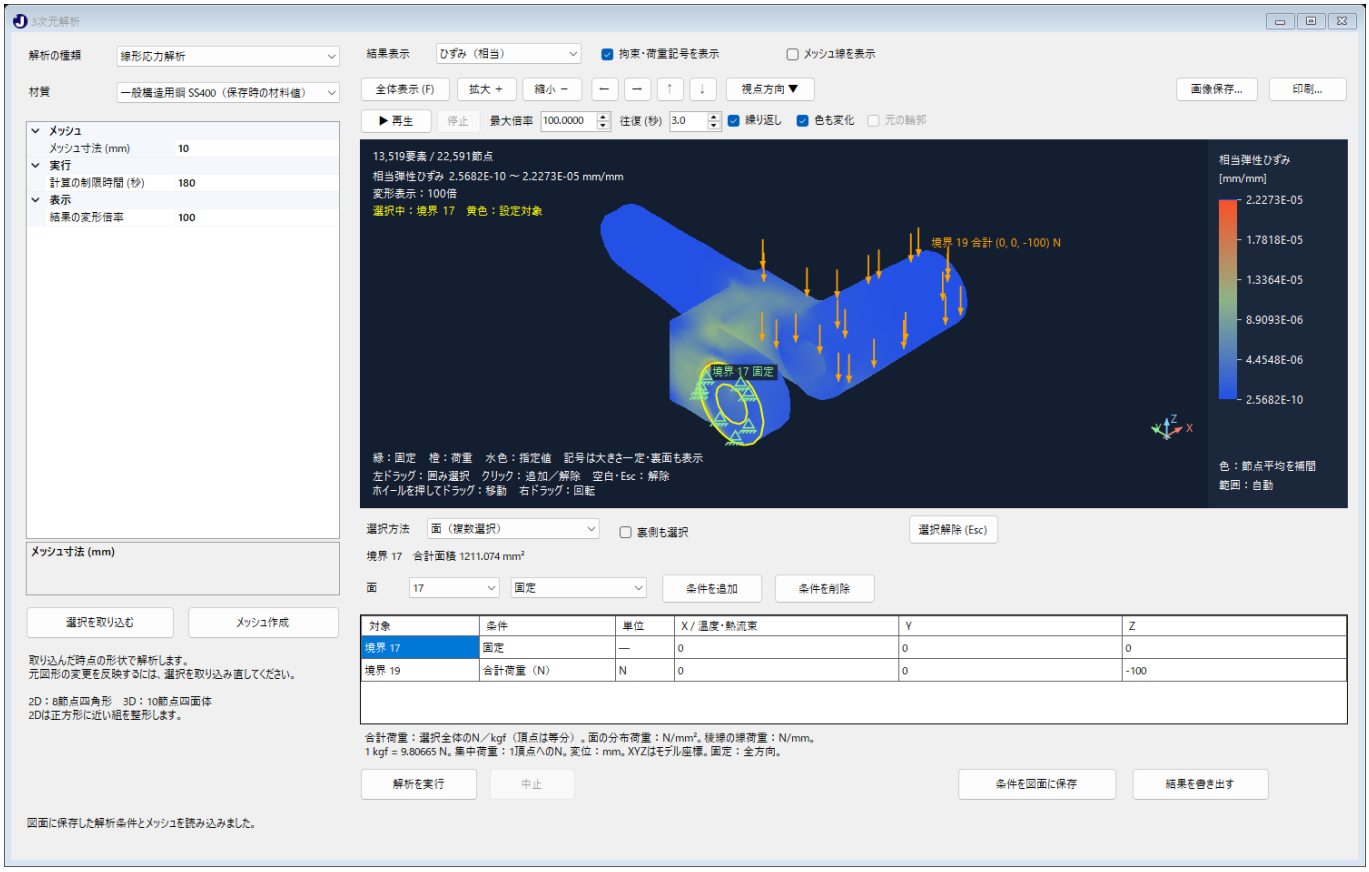


図 4-14 ひずみの分布表示例。凡例の成分と正負を確認します。

［ひずみ（相当）］は非負の相当弾性ひずみです。現在の等方線形弾性の定義では、ミーゼス応力をヤング係数で割った値に一致します。塑性ひずみではありません。

［ひずみ X・Y・Z］は垂直ひずみで、正が伸び、負が縮みです。2D の Z は板厚方向で、平面応力ではポアソン効果による厚み変化、平面ひずみでは 0 です。

［せん断ひずみ］は工学せん断ひずみ γ です。ひずみテンソルの非対角成分の 2 倍で、2D は XY、3D は XY・YZ・XZ を選びます。

単位は mm/mm で、0.001 は 0.1%です。［要素ひずみ CSV］には各成分と相当弾性ひずみを保存します。3D 内部の要素を含む数値は CSV で確認してください。

14 定常熱解析を行う

定常熱解析は、時間が十分に経って温度が変化しなくなった状態を求めます。形状の取り込みとメッシュ作成は応力解析と共通です。

1. ［解析の種類］で定常熱解析を選び、材料の熱伝導率を確認します。

2. 対象の面・辺などを選び、温度指定や熱流束、放熱の条件を追加します。

3. ［解析を実行］し、［温度］と［熱流束］を確認します。

熱条件	入力と働き
温度指定	℃。その場所を指定温度に維持
熱流束	W/m²。正は流入、負は流出
放熱 対流	熱伝達率 W/(m²K)と周囲温度 ℃
断熱	数値なし。その面や辺を一括放熱から除外

条件を付けない境界は初期設定で断熱です。定常解析では各独立領域に指定温度または放熱条件が必要です。全体が断熱で、1 点だけ 100℃に指定すると、十分時間が経った全体の温度は 100℃になります。周囲が自動的に 20℃になるわけではありません。

温度分布を試すには、長方形の左辺を 100℃、右辺を 20℃に指定し、残りを断熱にします。一定の熱伝導率なら、両辺の間に温度勾配が生じます。

3D では面に温度・熱流束・放熱・断熱を設定できます。稜線と頂点は温度指定に対応します。

15 非定常熱解析を行う

非定常熱解析は、初期温度から時間が経過する間の温度変化を求めます。材料の熱伝導率に加えて密度・比熱を使います。

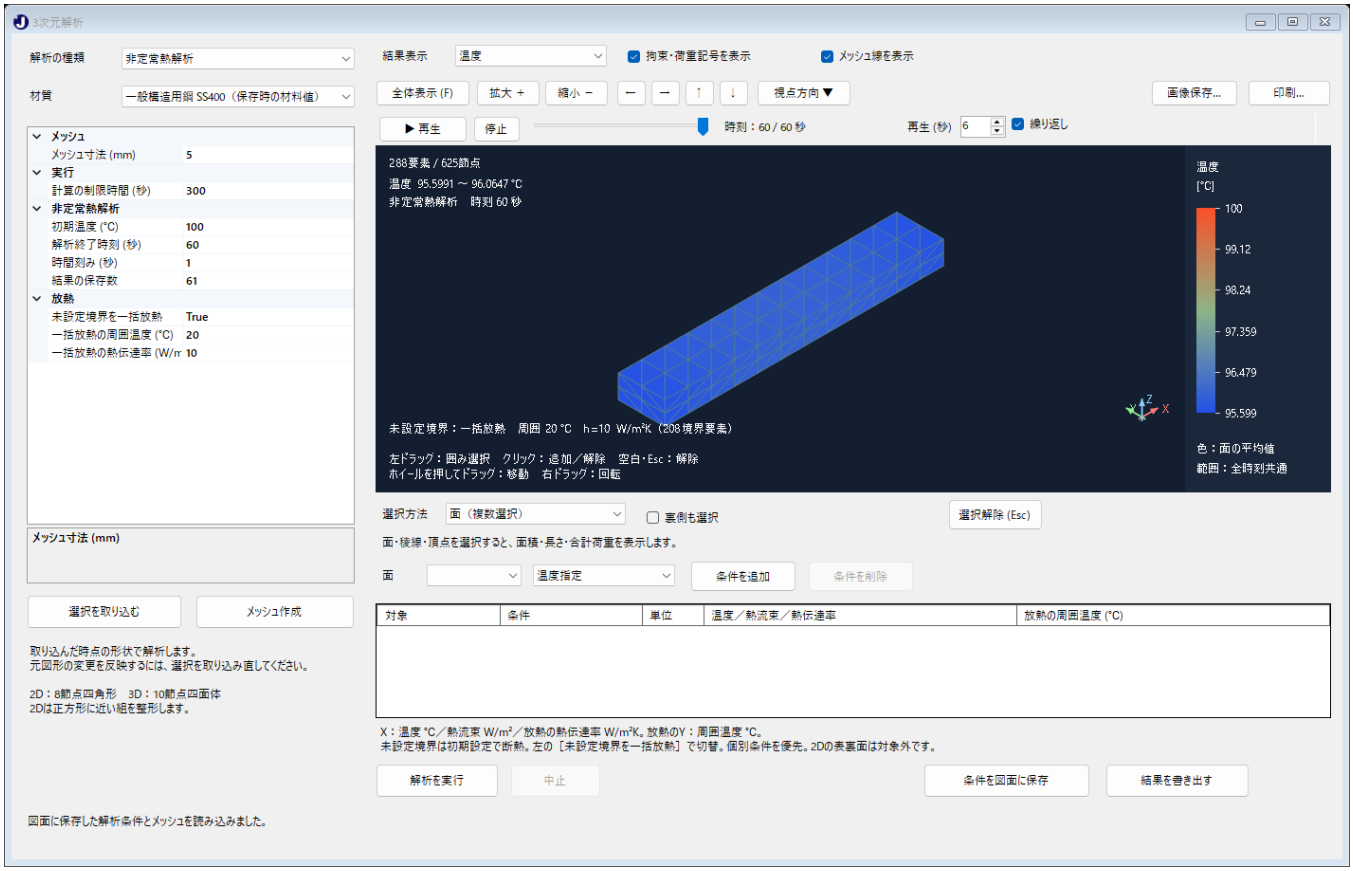


図 4-15 非定常熱解析の画面。左側に時間と放熱の設定、上部に時刻のスライダーがあります。

1. ［解析の種類］を［非定常熱解析］にします。
2. ［初期温度］［解析終了時刻］［時間刻み］を設定します。
3. 温度、熱流束、放熱などの条件を設定して［解析を実行］します。
4. スライダーで時刻を選ぶか、［再生］で温度の変化を確認します。

例えば初期温度 100°C、終了 60 秒、刻み 1 秒なら 100°Cから 60 秒後まで計算します。［計算の制限時間］は処理を待つ上限で、物理的な経過時間ではありません。

［温度指定］は、その場所を解析中ずっと同じ温度に保つ条件です。自然に冷える様子を見るだけなら、初期温度と放熱を設定し、温度指定は置きません。

16 未設定の境界を一括で放熱にする

左側の［放熱］にある［未設定境界を一括放熱］を True にし、［一括放熱の周囲温度］と［一括放熱の熱伝達率］を入力します。定常・非定常の両方で使用できます。

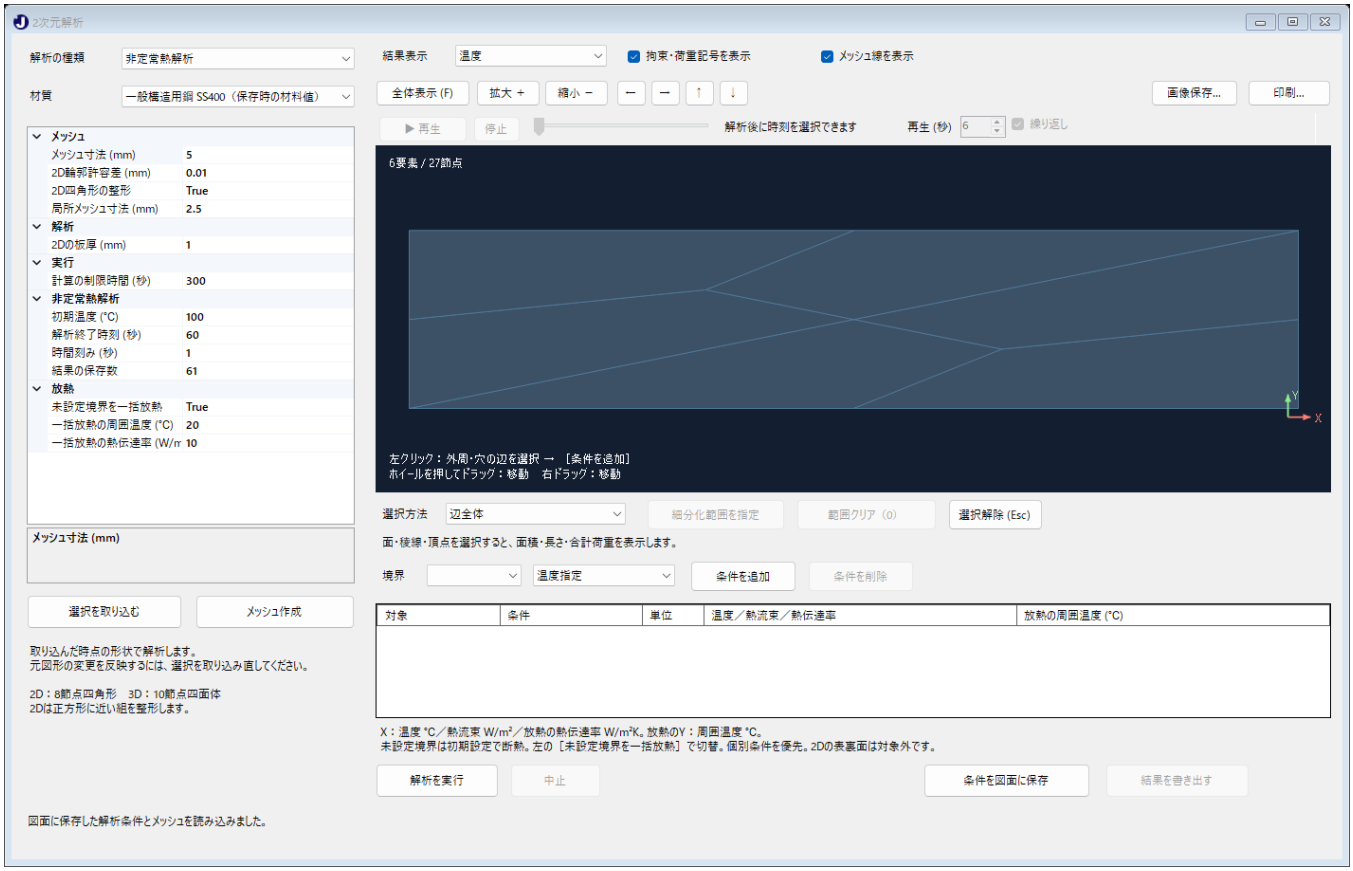


図 4-16 ［未設定境界を一括放熱］を True にした 2D の設定例。

適用する範囲

3D は外表面、2D は輪郭の辺の長さ×板厚に適用します。2D 板の表裏の広い面への放熱は含みません。

温度・熱流束・放熱・断熱を個別に設定した面や辺は一括設定から除外します。2D で辺の一部分を設定した場合は、その部分だけを除外します。稜線・頂点・節点への温度指定は、隣接する面や辺全体の放熱を除外しません。

熱伝達率の初期値 10 W/(m²K)は仮の値です。使用する流体や風速などに合わせて変更してください。周囲温度が部品より高ければ、対流条件は加熱側に働きます。

17 時刻別の結果を再生し保存する

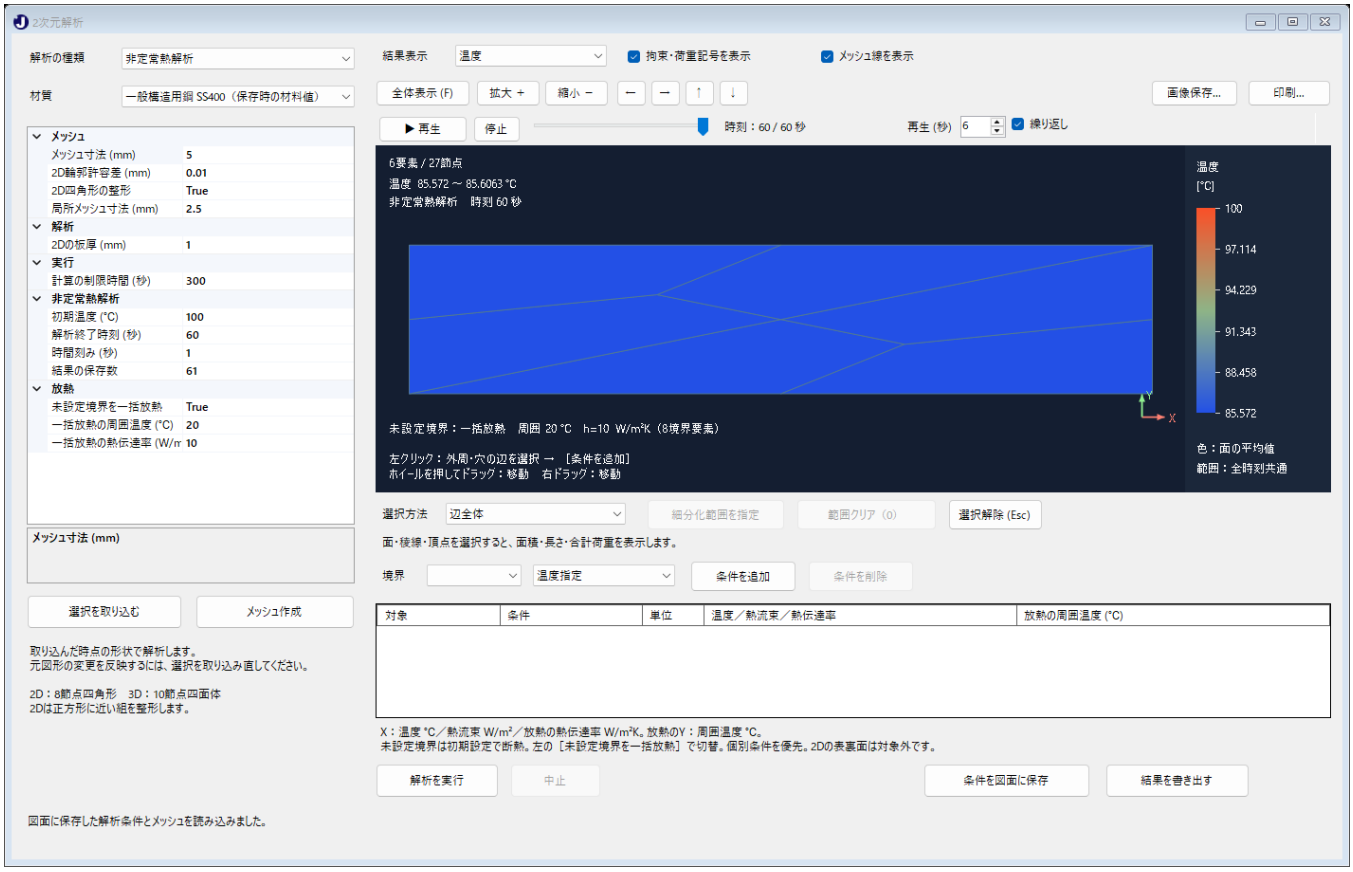


図 4-17 選択した時刻の温度分布。全保存時刻で共通の凡例を使います。

スライダーで時刻を切り替え、［再生］［一時停止］［停止］を操作します。熱解析の停止は開始時刻へ戻ります。［再生（秒）］は表示上の時間で、計算した物理的な時間を変更しません。

［結果の保存数］は開始・終了を含む保存時刻数の目安です。保存数を減らしても計算の時間刻みは変わりません。大きなメッシュではメモリー上限に合わせて保存数を減らします。

［熱流束］も選択した時刻の温度から表示します。画像保存・印刷はその時刻で一時停止し、時刻と放熱設定も出力します。JSON は全保存時刻、CSV は時刻別の節点温度を書き出せます。

適用範囲

材料値と境界条件は解析中一定です。時間で温度や熱流束を切り替える表、放射、相変化、温度に依存する物性、流体解析、熱応力連成は含みません。時間刻みを小さくした結果と比較して、刻みの影響を確認してください。

18 梁計算を行う

［解析］→［2 次元解析］または［3 次元解析］→［梁計算］を開きます。全長で材料・断面が共通の直線梁を扱います。

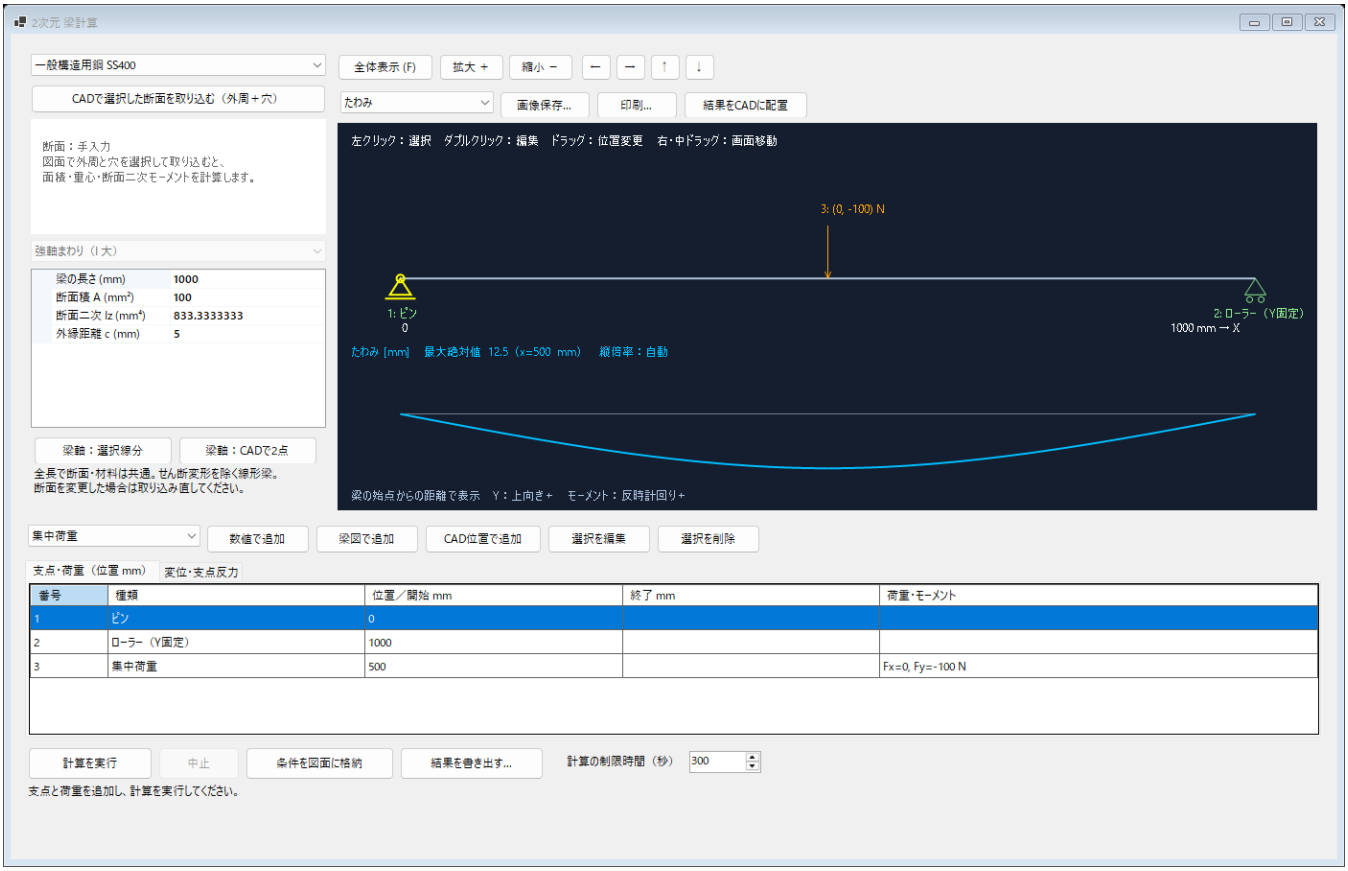


図 4-18 2 次元梁計算の画面例。断面、支点・荷重、結果を確認します。

1. ［CAD で選択した断面を取り込む］から平面の面、または外周と穴を選びます。未選択なら CAD 側で選択し、Enter で確定します。
2. 梁軸を線分または CAD の 2 点から取り込みます。長さの直接入力もできます。
3. 支点・荷重の種類を選び、数値、梁図、CAD 位置のいずれかで追加します。分布荷重は開始・終了の 2 点を指定します。
4. ［計算を実行］し、たわみ、せん断力、曲げモーメント、垂直応力などを切り替えます。

2D は Y の上向きが正で、下向きの荷重は負です。集中荷重は N、分布荷重は N/mm、モーメントは N·mm です。支点・荷重は表や梁図のダブルクリックで編集できます。

断面を手入力する場合は面積 A、断面二次モーメント I、外縁距離 c を設定します。c＝0 では曲げ応力を評価しません。

19 3D 梁の座標と断面

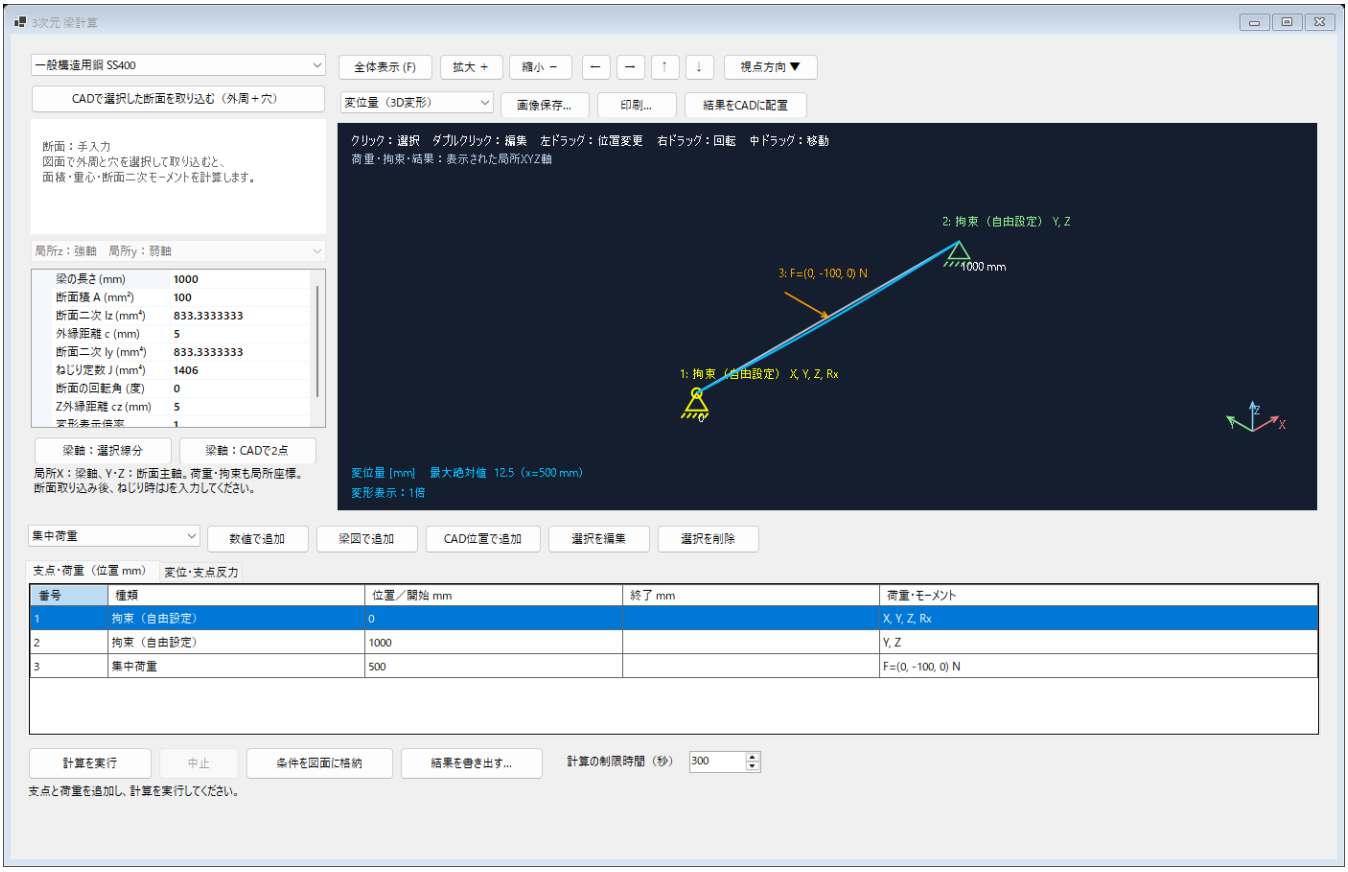


図 4-19 3 次元梁計算の画面例。断面、支点・荷重、結果を確認します。

3D 梁では、X は梁の始点から終点、Y・Z は断面の主軸です。荷重と拘束は画面に示す局所座標で入力します。［断面の回転角］で梁軸まわりの向きを調整できます。

拘束は 6 方向の自由・固定を指定します。ねじりの剛体回転を防ぐ拘束も必要です。モーメントの正方向は各軸の右ねじ方向です。

CAD 断面から面積・重心・断面二次モーメントなどを求めますが、ねじり定数 J は自動推定しません。Mx を与える場合は、断面に合う J を入力してください。J を Iy＋Iz で代用する設定ではありません。

2D・3D とも微小変形の線形弾性梁で、せん断変形、塑性、座屈、大変形、途中で断面が変わる梁は扱いません。垂直応力は軸力と曲げによる値で、ミーゼス応力ではありません。

［結果を CAD に配置］でグラフなどを図面に配置できます。3D 変位量モードは変形曲線、それ以外は作業平面上の結果グラフを配置します。

20 トラスとラーメン

CAD で線分または直線ポリラインを選び、［トラス］または［ラーメン］へ取り込みます。トラスは軸力を伝えるピン接合、ラーメンは曲げを伝える剛接合の部材です。

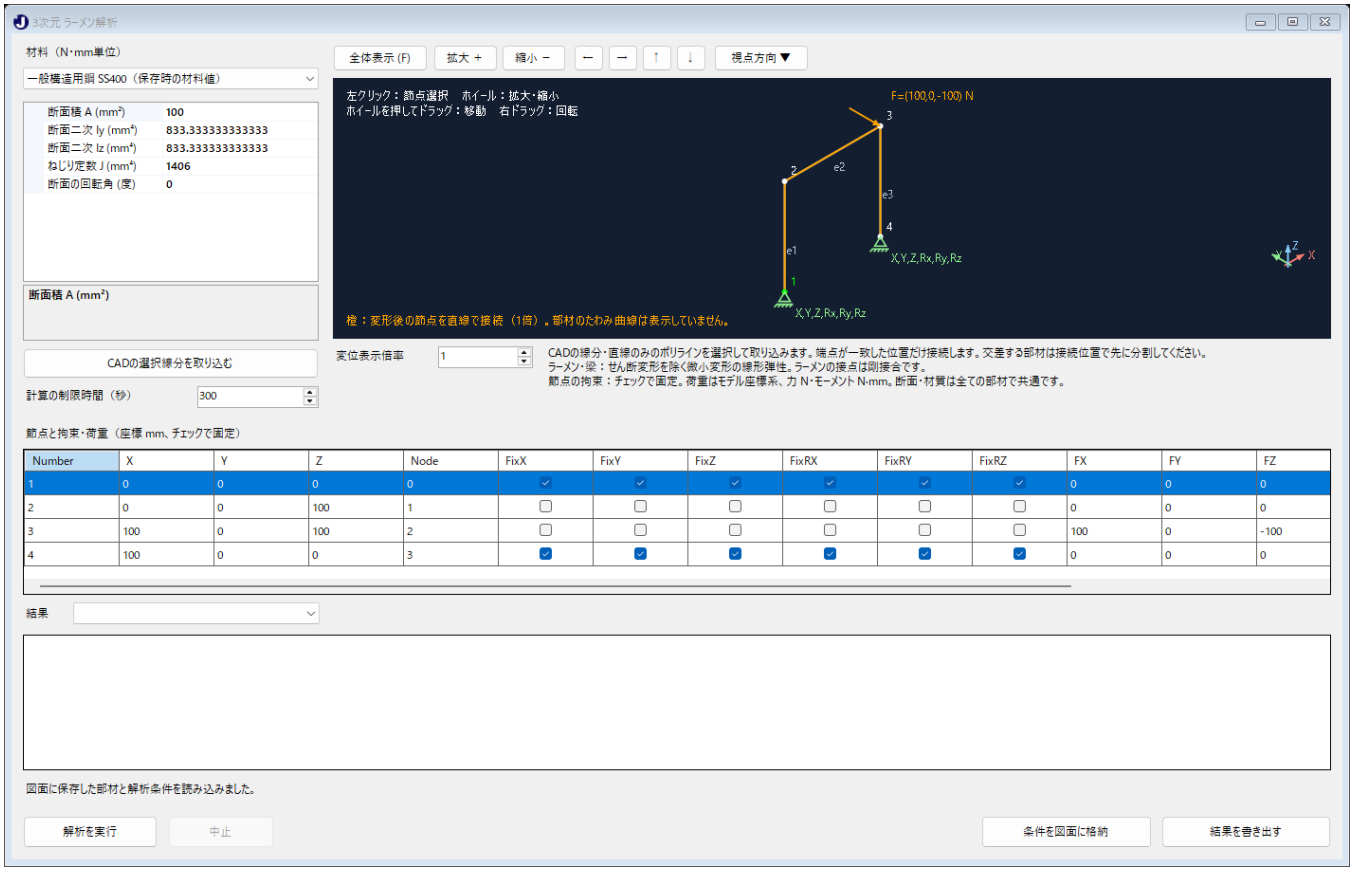


図 4-20 3D ラーメンの節点と条件の画面例。チェックした自由度を固定します。

1. 部材を取り込み、材料と共通の断面値を設定します。
2. 節点をクリックして表の行を選び、固定する方向と荷重を入力します。
3. ［解析を実行］し、節点変位・反力と部材力を確認します。

端点が一致する部材を接続します。途中で交差するだけの線は接続しないため、接続したい位置で先に分割してください。2D はモデル XY 平面と平行な同一平面に配置します。

変形表示は変形後の節点を直線で結ぶ表示で、部材内のたわみ曲線ではありません。全ての部材で材料・断面値を共通にし、2400 自由度を上限とします。

21 画像と印刷と数値を出力する

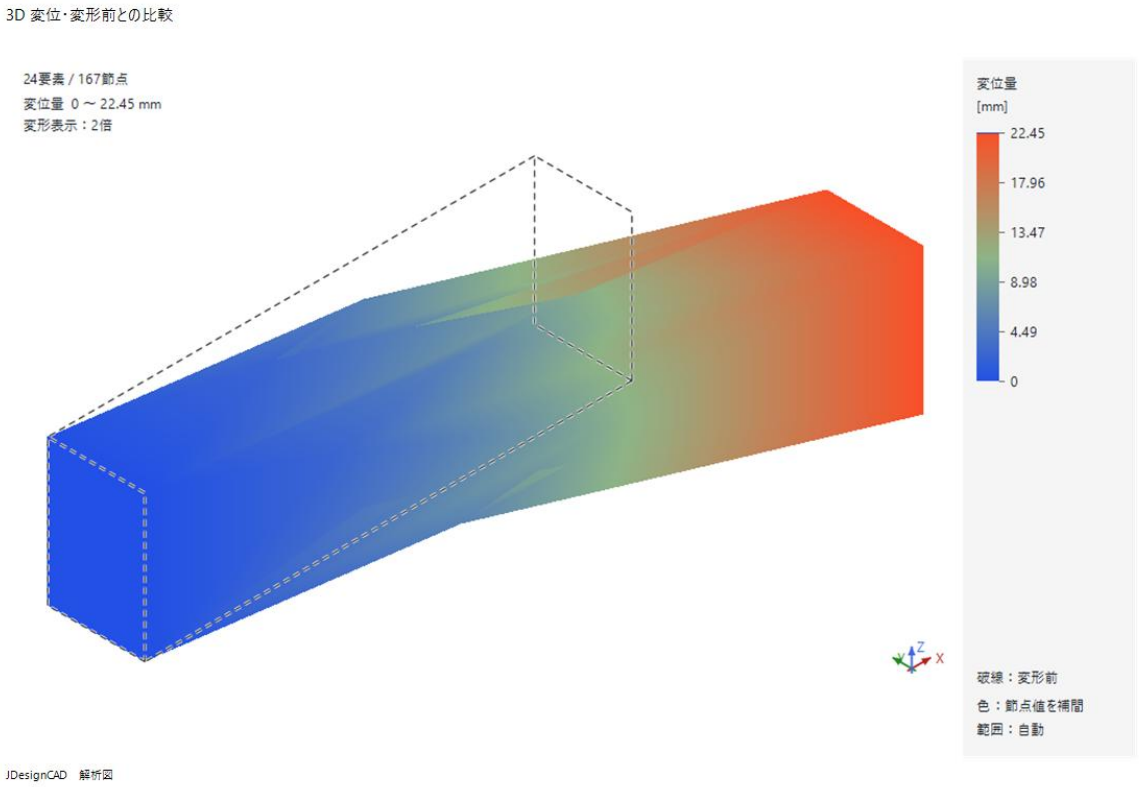


図 4-21 白背景の印刷ページの例。図形の向き、結果、凡例、変形倍率などを出力します。

［画像保存…］は現在の表示を PNG または JPEG へ保存します。［印刷…］は白背景のプレビューを開き、用紙とプリンターを選んで印刷します。プレビューを開くだけでは印刷しません。

全体を載せる場合は先に［全体表示］を押し、メッシュ線・記号・元の輪郭を設定してください。アニメーション中は一時停止して、そのフレームを出力します。選択中の黄色表示と操作案内は画像出力に含めません。

［結果を書き出す］は数値データ用です。JSON のほか、結果に応じて節点反力、要素ひずみ、時刻別温度などの CSV を選べます。CSV の節点・要素番号は 1 から始まります。

22 条件を図面に保存する

解析画面で［条件を図面に保存］または［条件を図面に格納］を押し、その後で通常の JDC 保存を実行します。解析画面のボタンだけではファイル保存は完了しません。

材料値、メッシュまたは部材、境界条件、非定常熱の時間・放熱設定などを保持します。図面を開き直した後は再計算して結果を得ます。数値結果を残す場合は、別途［結果を書き出す］を使用してください。

変更した項目	次に必要な操作
CAD 側の形状	取り込み直し、メッシュ作成、条件の再設定
メッシュ寸法や輪郭許容差	メッシュを作り直す
材料、境界条件、物理設定	解析を再実行
表示項目、変形倍率、視点	表示だけ変更。再解析は不要
非定常熱の初期温度や時間刻み	解析を再実行

条件は 2D・3D、解析種類ごとに一組です。熱解析は定常・非定常で条件を共用するため、両方の条件を独立して残したい場合は JDC を別名で保存します。

計算できないとき

まず閉じた形状か、材料値と単位が正しいか、拘束不足や矛盾する温度指定がないかを確認します。［中止］または計算制限時間で処理を止められます。形状を簡略化した場合は、元の評価対象を変えていないかも確認してください。

鋭い角や点・稜線に集中した荷重の近傍では、最大応力がメッシュに強く依存することがあります。メッシュを細かくした結果と比較し、最大値だけで結論を出さないでください。

23 リブと穴のある部品の解析例

練習 16.jdc のソリッドを使った 3D 解析例です。細長い部品に複数の穴と補強リブがあり、単純な梁とは異なる形状の変形や分布を確認できます。

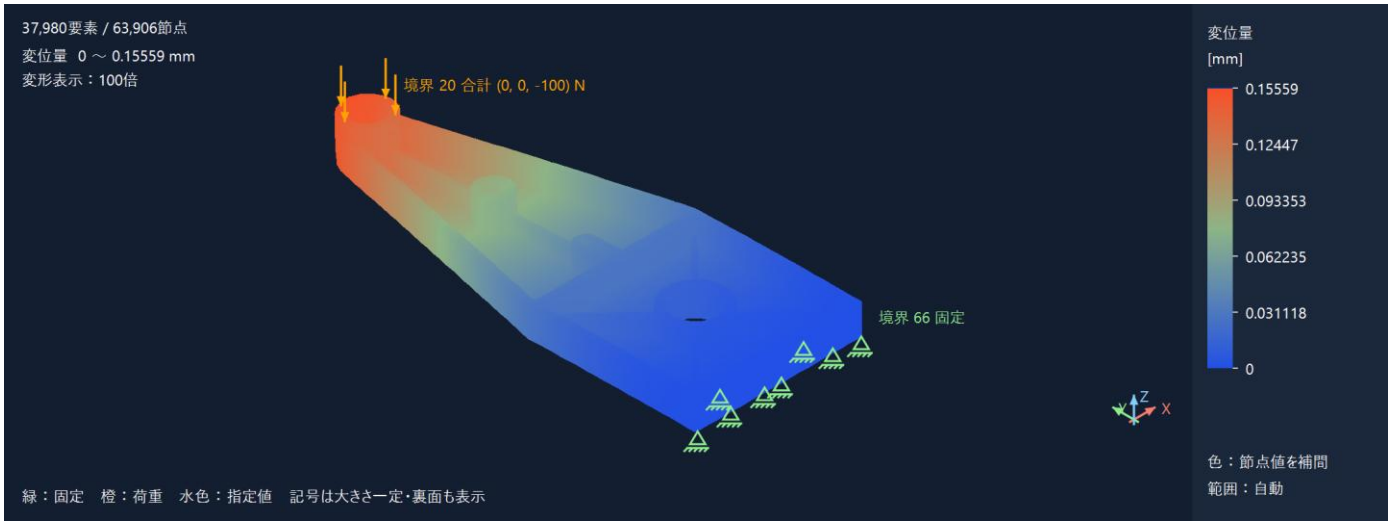


図 4-22 練習 16 の変位表示。変形は 100 倍で表示しています。

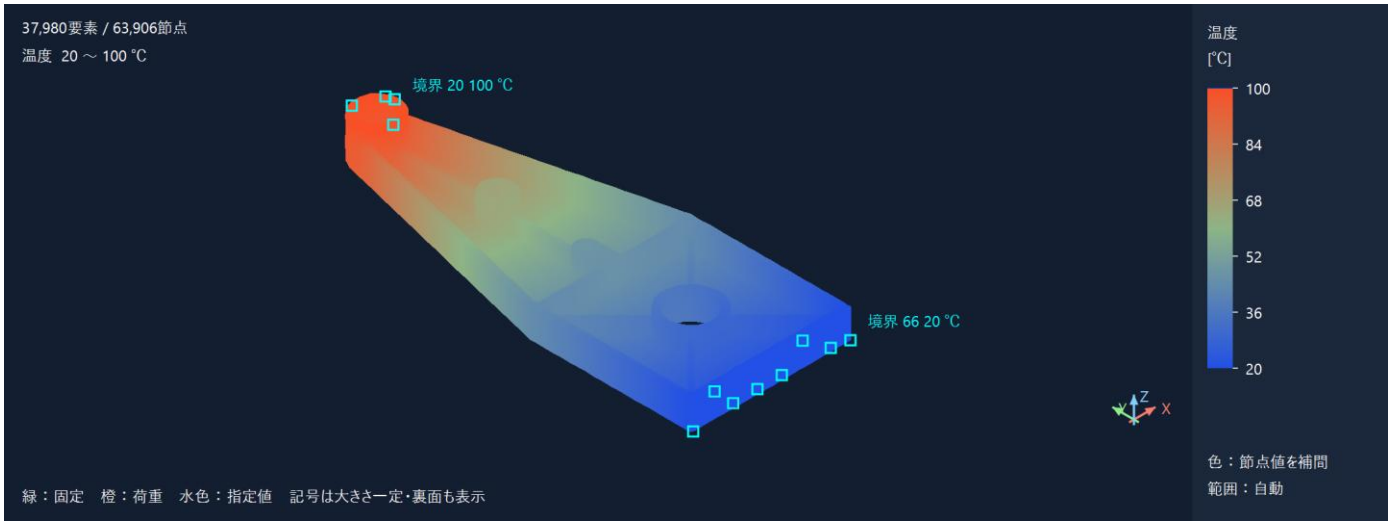


図 4-23 同じ形状の定常温度分布。指定温度 20°Cと 100°Cの間に温度勾配が生じます。

機械的な条件は SS400、メッシュ寸法 10mm、Y 最小側の端面を固定、先端側の上面に Z 方向の合計荷重 −100N です。熱解析では固定に使った面を 20°C、荷重に使った面を 100°Cとし、残りを断熱としています。

操作と表示を説明するための条件です。実際の取付面、荷重、材料、伝熱条件に合わせて設定し直してください。

24 フランジと曲面を含む部品の解析例

練習 8.jdc と練習 13.jdc を使い、フランジ・穴・曲面を含む形状の結果を表示します。結果の種類を切り替えても、同じ条件で計算した変位・応力・ひずみ・反力を確認できます。

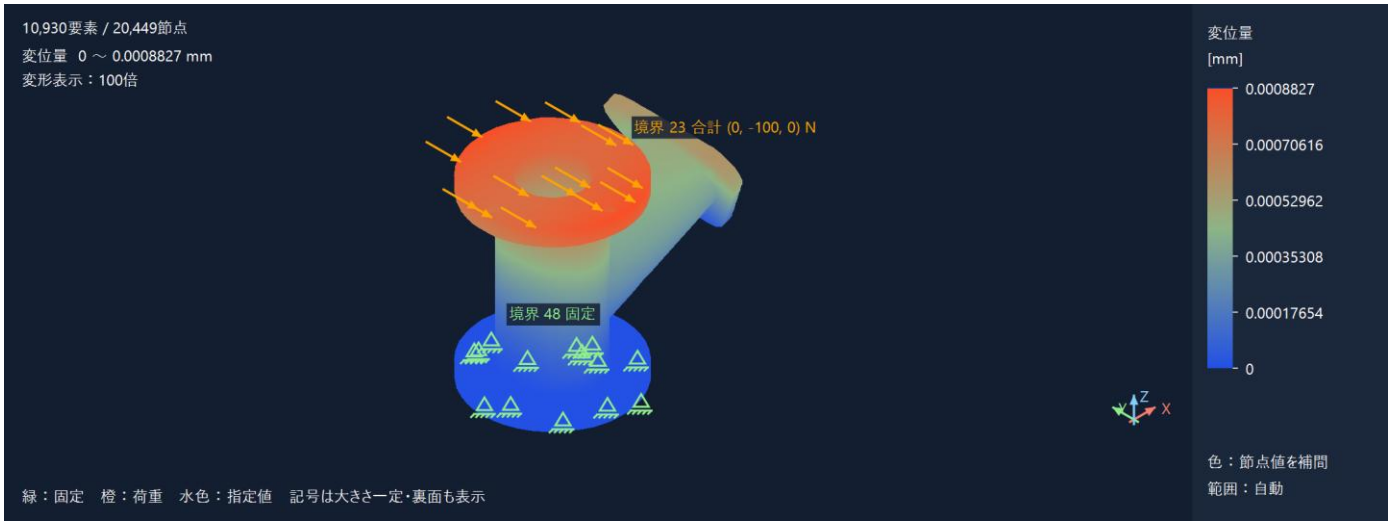


図 4-24 練習 8 の変位表示。Z 最小側の面を固定し、Z 最大側の面に Y 方向の合計荷重－100N を設定。

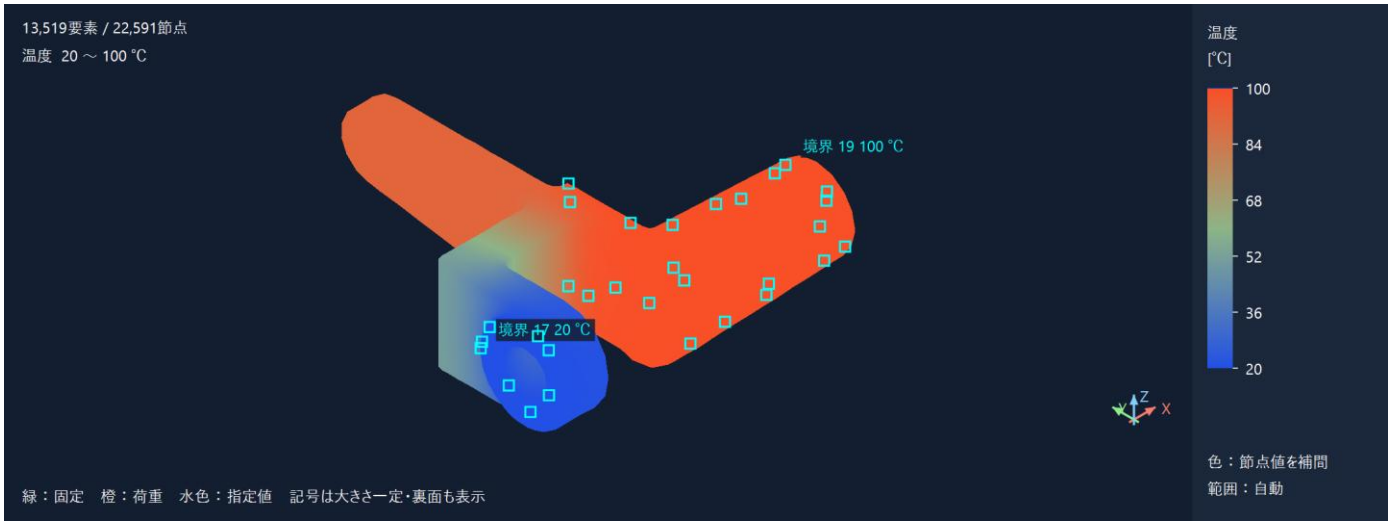


図 4-25 練習 13 の定常温度表示。X 最小側の端面を 20°C、X 最大側の端面を 100°Cとし、残りを断熱。

両例とも材質は SS400、メッシュ寸法は 10mm です。図 4-24 の変形は 100 倍表示です。画像には結果の単位と色の範囲を残し、図形の色だけで大小を比較しないでください。

設計検討では条件の妥当性を確認し、必要に応じてメッシュを細かくした結果との比較を行ってください。

25 目的に合う解析を選ぶ

解析は STD・Pro で利用できます。まず知りたい量と必要な形状を決めます。単純な棒のたわみを調べる場合と、穴の周囲の応力を調べる場合では適したモデルが異なります。

確認したいこと	選ぶ機能	用意するもの
薄板の面内の変形・応力	2D 応力解析（平面応力）	閉じた輪郭／面、板厚
奥行方向の変形を拘束した断面	2D 応力解析（平面ひずみ）	適切な平面ひずみの仮定
厚い部品・穴・複雑な形の応力	3D 応力解析	閉じたソリッド、材料、接合
直線梁のたわみ・曲げモーメント	2D／3D 梁計算	軸線、断面、支点と荷重
ピン接合した骨組の軸力	トラス	接続する直線部材
剛接合した骨組の曲げ・軸力	ラーメン	直線部材と断面性能
十分時間が経過した温度分布	定常熱解析	材料、温度・熱流束・放熱
冷却・加熱の時間経過	非定常熱解析	上記＋初期温度・時間・密度・比熱

モデル化の限界

面・ソリッドの構造解析は微小変形の線形弾性です。2D 平面応力は薄板の「面内」の問題で、板の面外曲げを解析する板要素ではありません。梁は細長い直線部材の近似で、穴縁などの局所応力は求めません。

塑性・大変形・座屈・振動・離間や滑りの接触・熱応力連成・自重の自動荷重化・設計基準の合否判定は対象外です。変形アニメーションは計算済みの結果を補間する表示で、振動解析ではありません。

最初は簡単な形で荷重方向と拘束を確かめ、必要な細部を追加します。最大値だけでなく、変形の向き、荷重と反力の釣り合いも確認します。

26 解析条件を組み立てる順序

解析前に記録する項目

項目	決める内容	例
目的	比較する値と場所	先端変位、穴縁の応力、60 秒後の温度
形状	残す細部と省略する細部	評価する穴と丸みは残す
材料	実製品に対応する物性	E、ポアソン比、熱伝導率など
支持・接合	どこが何に固定されるか	取付面固定、完全接合
外部作用	大きさ・方向・範囲	面全体に合計－100 N
単位	長さ、荷重、物性の単位	mm、N、N/mm²
精度確認	メッシュ／時間刻みの比較	寸法を小さくして変化を記録

共通の操作順

1. 解析用の図形を選び、画面の［選択を取り込む］を押します。元 CAD 形状のコピーが解析対象になります。

2. 材料、解析の種類、2D なら板厚、複数ソリッドなら材料と接合を設定します。

3. ［メッシュ作成］を実行し、形状の欠落、穴、細部、接合を確認します。

4. 境界を選び、条件表へ固定・荷重または温度・放熱などを追加し、数値を確定します。

5. ［解析を実行］後、分布と数値を確認します。必要ならメッシュを細かくし、条件を設定し直して比較します。

6. 条件を図面へ格納し、JDC を保存します。画像と数値結果は必要に応じて別途書き出します。

CAD 側を変更しても取り込み済みの解析形状は自動更新されません。形状変更後は取り込み直します。表示倍率・視点・色の切替は物理条件を変更しません。

27 複数材料と完全接合を設定する

解析対象として、面で接する 2 個のソリッドを別部品のまま用意します。例は各 40×20×10 mm で、接合面積は 20×10＝200 mm²です。3D 解析を開き［選択を取り込む］を実行します。取り込んだ時点の形状が解析対象です。元図形を変えた場合は取り込み直します。

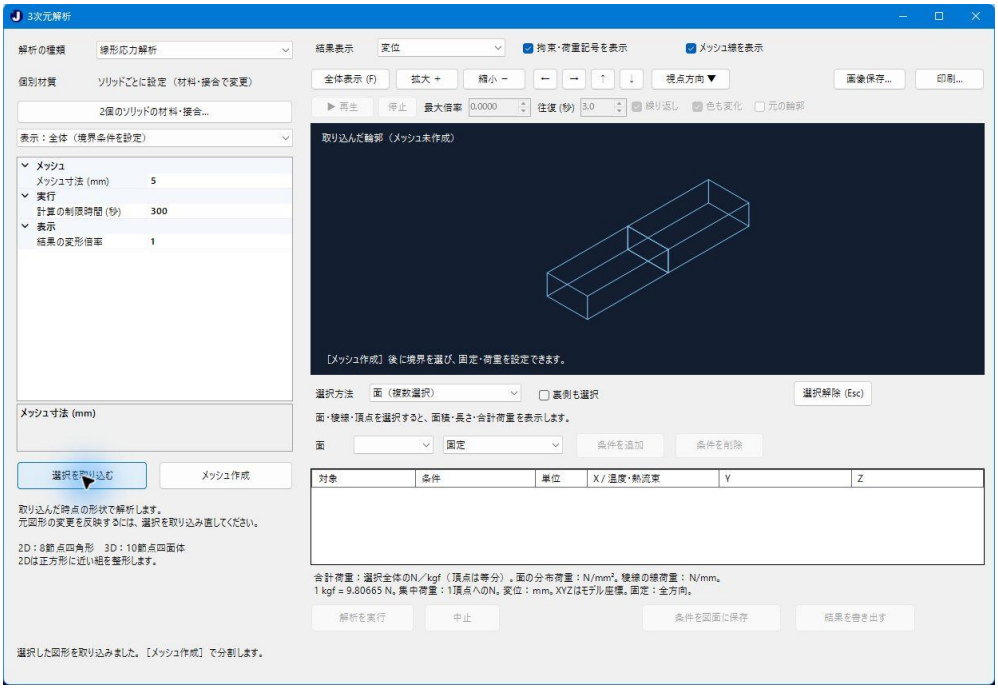


図 204　2 部品を取り込んだ 3D 解析画面

ソリッドごとに材料を選ぶ

［2 個のソリッドの材料・接合…］を開き、［ソリッドの材料］で各行の材料を一覧から選びます。左の行を選ぶと右のプレビューで対応する部品が強調されます。材料名だけでなく、材質リスト登録でヤング係数・ポアソン比・密度などの値と単位を確認します。図の鋼とアルミニウムは操作例です。実設計では対象材料の条件に合う値を設定してください。

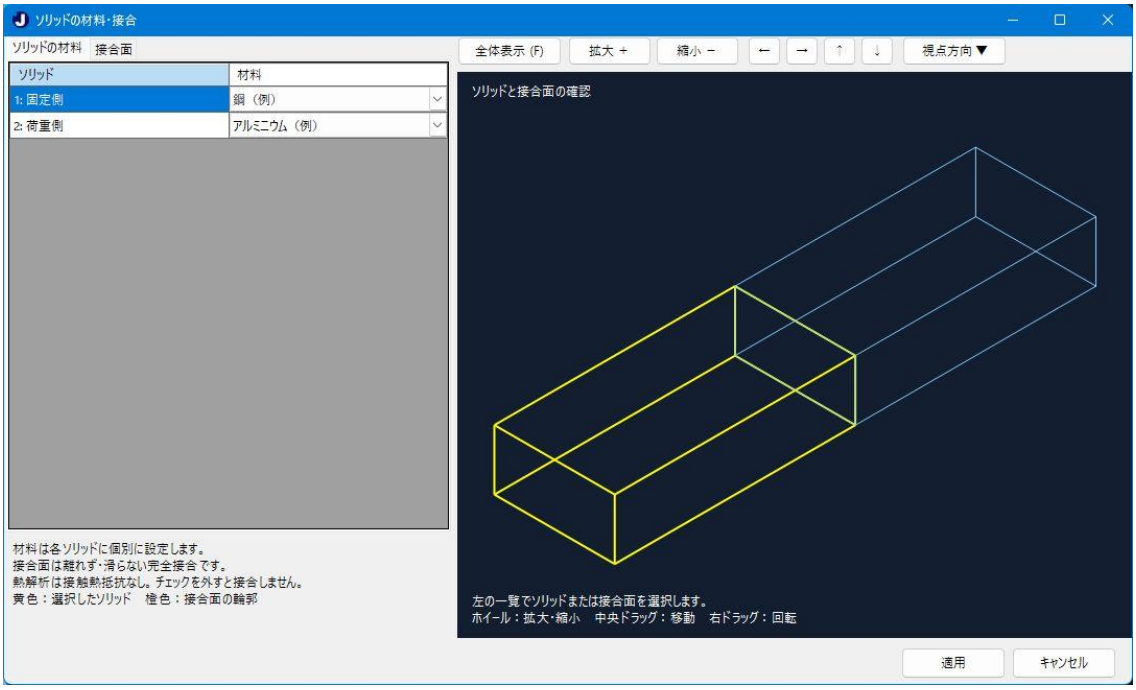


図 205　材料を選ぶ行と、ハイライトされる対象部品

接合面を確認して適用する

［接合面］へ切り替え、接するソリッドの組合せと面積を確認します。行を選ぶと対応する 2 部品と接合輪郭を表示します。［完全接合］が有効な面は変位・温度を連続させる接合です。摩擦、すべり、剥離や接触後の離間を扱う非線形接触解析ではありません。必要な接合を選び［適用］します。

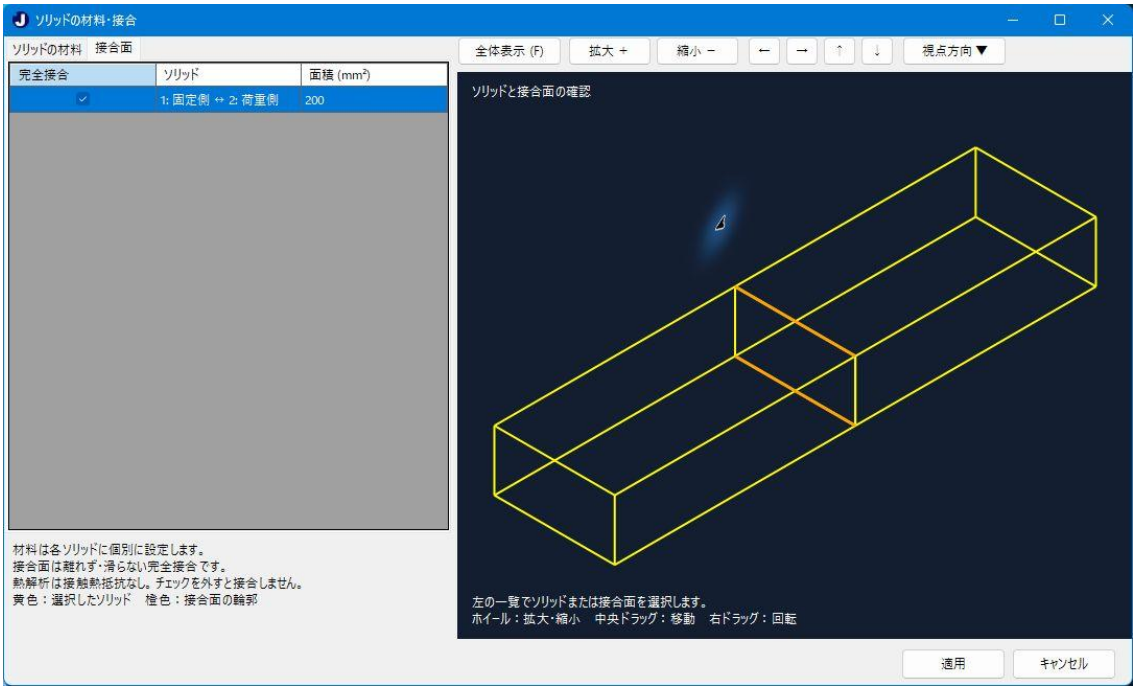


図 206　200 mm²の接合面を確認する

続いて［メッシュ作成］を実行し、表示を［全体（境界条件を設定）］に戻して固定・荷重などを設定します。材料を変更したら再解析します。接合の有効・無効を変更した場合はメッシュも作り直し、境界条件を設定し直します。個別部品の表示は形状を確認するために使います。

面で接しない部品や点・稜線だけで接する部品は、この完全接合で一体化しません。独立部品にはそれぞれ必要な拘束が必要です。保存済み解析に元ソリッドが含まれない場合、接合を変えるには元の形状を取り込み直します。解析条件を図面へ保存し、計算条件・メッシュ寸法と結果を一緒に確認してください。

28 メッシュ寸法と計算精度

メッシュ寸法を小さくすると細部を表現しやすくなりますが、要素・節点が増え、計算時間とメモリーも増えます。最初から極端に細かくせず、目的の値が安定するかを段階的に確認します。

2D と 3D の違い

項目	2D	3D
新規メッシュ	8 節点四角形	10 節点四面体
形状	一つの外周と穴、平面の面	閉じたソリッド
局所細分化	矩形範囲を最大 32 個	本版の局所細分化対象外
曲面・曲線	境界を近似して分割	表面を三角形で近似

2D 四角形の整形

［2D 四角形の整形］は既定で True です。穴のない長方形は辺に沿った格子にします。長方形では最終四角形の辺長が入力メッシュ寸法のおよそ 1/6 を目安とするため、入力値と実際の要素寸法を同一と考えないでください。要素数と表示を確認します。

穴・曲線・切欠きがある形は、その形状を保って分割します。整形の比較は False にしてメッシュを作り直します。どんな荷重でも整形によって精度が上がるという意味ではありません。

比較のしかた

1. 材料・支持・荷重を固定し、メッシュ寸法 h で解析します。

2. 次に h より小さい寸法で作り直し、同じ場所に同じ合計の条件を設定します。

3. 同じ箇所の変位、評価範囲の応力、合計反力、要素数を記録して比較します。

4. 差が大きい場合はさらに確認します。角・点荷重・固定端近傍で最大応力だけが増え続ける場合は、条件と形状の理想化も見直します。

再メッシュすると節点・辺区間番号が変わるため、旧番号をそのまま流用せず、条件を設定し直します。

29 荷重の入力を間違えない

合計荷重、分布荷重、線荷重、集中荷重は異なる量です。同じ「100」でも作用する力が変わります。

条件	単位	入力例と総荷重
合計荷重	N／kgf	2 面まとめて 100 N なら全体で 100 N
分布荷重	N/mm²	面積 200 mm²に 0.5 なら 100 N
線荷重	N/mm	長さ 20 mm の稜線に 5 なら 100 N
集中荷重	N	一つの節点へ 100 N

合計荷重は面では面積、稜線では長さ、複数頂点では点数で配分します。2D の分布荷重に使う面積は、選択した辺の長さ×板厚です。1 kgf＝9.80665 N です。

入力例：下向きに合計 100 N

2D で下向きが－Y なら、合計荷重の X＝0、Y＝－100 とします。3D で下向きが－Z なら、X＝0、Y＝0、Z＝－100 です。座標軸を画面で確認して入力します。

面・ソリッドの成分荷重は一定方向の力です。曲面全体で各点の法線方向へ変わる圧力として自動解釈しません。矢印の長さも力の大きさを定量表示するものではありません。

拘束の確認

固定は全並進成分を 0、変位指定は全並進成分を指定します。連続体で任意の一軸だけを自由にする設定とは区別します。梁・トラス・ラーメンの自由度チェックは、それぞれの画面の規則に従います。

一直線の稜線だけの固定では、その線を軸とする剛体回転が残ることがあります。接合されていない部品も独立して動けます。制限時間を増やす前に、実際の取付状態と同じ拘束になっているか確認します。

同じ箇所に荷重行を重ねると加算します。条件を置き換える場合は古い行を削除してください。

30 実習：梁計算を既知の値と比較する

長さ 1,000 mm、幅 10 mm・高さ 20 mm の矩形断面を持つ両端支持梁に、中央で下向き 100 N を加える例です。数値は操作と単位の確認用で、製品の許容値ではありません。

入力項目	値
ヤング係数 E	200,000 N/mm²
梁長 L	1,000 mm
断面積 A	200 mm²
断面二次モーメント I	6,666.6667 mm⁴
中立軸から外縁まで c	10 mm
始点の支点	ピン、位置 0
終点の支点	ローラー（Y 固定）、位置 1,000
集中荷重	位置 500、Fx＝0、Fy＝－100 N

1. ［解析］→［2 次元解析］→［梁計算］を開きます。初期条件と重複しないよう支点・荷重表を確認します。

2. 材料の E と断面値を確認し、上表の値を入力します。CAD 断面を使う場合は、高さ 20 方向の曲げに対応する軸を選びます。

3. ［計算を実行］し、たわみ・曲げモーメント・垂直応力・支点反力を確認します。

理論式による確認値

Euler–Bernoulli 梁の式では、中央たわみの大きさ $PL^3/(48EI)$＝1.5625 mm、最大曲げモーメント PL/4＝25,000 N·mm、最大曲げ応力 Mc/I＝37.5 N/mm²です。支点反力は上向きに各 50 N です。

計算した変位の符号は下向きなら負ですが、「変位量」の大きさ表示は非負です。結果が大きく違う場合は、I の曲げ軸、E の桁、mm⁴、荷重の重複、支点位置を確認します。3D ソリッドの局所応力と、この梁理論の値をそのまま同一視しないでください。

31 3D 梁の局所軸・ねじり定数

3D 梁では、CAD 上の方向が斜めでも、画面に表示する梁の局所軸で荷重・拘束・結果を扱います。X は始点から終点、Y・Z は断面の主軸です。

1. CAD で平面の断面または外周と穴を選び、［CAD で選択した断面を取り込む］を実行します。

2. ［梁軸：選択線分］または［梁軸：CAD で 2 点］で軸を指定します。

3. 断面の強軸・弱軸と、［断面の回転角］を確認します。回転すると荷重・拘束の基準となる局所軸も回転します。

4. 固定端または［拘束（自由設定)]、集中荷重・モーメント・区間分布荷重を追加します。

5. ねじり荷重がある場合は、断面に合ったねじり定数 J を入力して計算します。

J は自動計算されない

CAD 断面の取り込みで面積・重心・断面二次モーメントを計算しますが、ねじり定数 J は 0（未設定）になります。Mx 荷重がある場合は J が必要です。一般の断面で J＝Iy＋Iz と置き換えないでください。

量	単位・方向
Fx・Fy・Fz	N、局所 XYZ
Mx・My・Mz	N·mm、各軸の右ねじ方向が正
区間分布荷重	N/mm、開始・終了位置と方向
Iy・Iz・J	mm^4
拘束	X・Y・Z・Rx・Ry・Rz を確認

ねじり方向に自由な剛体回転が残らないよう、必要な Rx 拘束を設定します。3D 梁は軸力、2 方向の曲げ、ねじりを扱いますが、せん断変形・反り拘束・断面の途中変更は扱いません。垂直応力にはねじりせん断応力を含みません。

32 トラス・ラーメンの接続と結果

トラスは軸力だけを伝えるピン接合部材、ラーメンは曲げも伝える剛接合の梁要素です。見た目が同じ骨組でも、接合の仮定で結果は変わります。

取り込み前の作図

1. 部材の軸線を線分または直線ポリラインで描きます。2D はモデル XY と平行な同一平面へ置きます。

2. 接続したい位置で線分を分割し、端点を一致させます。途中で交差するだけの線分は接続されません。

3. 軸線を選んでトラス／ラーメン画面へ取り込み、節点番号と接続を確認します。

4. 材料と断面値を設定し、節点の拘束・荷重を入力して計算します。チェックした自由度は固定、未チェックは自由です。

項目	トラス	ラーメン
伝達する主な力	軸力	軸力・せん断・曲げ、3D はねじり
回転自由度	なし	あり
主な結果	節点変位・反力、軸力・軸応力	節点変位・反力、部材端力
変形表示	変形後の節点を直線接続	同左。部材内のたわみ曲線ではない

力は N、モーメントは N·mm で、節点への入力はモデル座標です。3D 梁単体の局所座標入力と混同しないでください。部材力の i/j は部材の始点／終点で、部材局所座標の値です。

設定と規模の制約

材料・断面積・断面二次モーメント・ねじり定数は全ての部材で共通です。自由度の上限は 2,400 です。異なる断面を部材ごとに自動配分する設計機能ではありません。

最大節点変位は節点での値です。部材中央のたわみや断面の局所応力が必要な場合は、梁計算または適切なソリッド解析も検討します。

33 定常熱解析の境界条件を読む

定常熱解析は、温度が時間で変わらなくなった状態を求めます。初期温度からの冷却過程を知りたい場合は非定常熱解析を選びます。

条件	入力	意味
温度指定	X：°C	選んだ場所をその温度に維持
熱流束	X：W/m²	正で流入、負で流出
放熱（対流）	X：W/(m²K)、Y：周囲温度°C	表面と周囲の温度差に応じた熱交換
断熱	数値なし	熱の出入りを 0 にする

未設定の境界は、［未設定境界を一括放熱］が False なら断熱です。周囲が自動的に 20°Cになるわけではありません。各独立領域に温度指定または有効な放熱条件など、温度の基準が必要です。

実習：左右の温度差を与える

長方形の左辺を 100°C、右辺を 20°C、残りを断熱にし、一定の熱伝導率を設定します。メッシュを作成して辺を選び、二つの温度指定を登録して計算します。左右の間に連続した温度勾配が生じることを確認します。

一方の 1 節点だけを 100°Cに固定し、残り全てを断熱にすると、定常状態で全体が 100°Cになる場合があります。これは冷却途中を示す計算ではありません。

複数材料の熱接合

完全接合面は熱抵抗なしでつながります。有効な接合面は一括放熱の対象から除外します。接合を外した面は外表面として扱うので、一括放熱を有効にした場合はその面にも作用します。放熱させたくない面には個別に断熱を設定します。

2D の対流は輪郭辺の長さ×板厚に作用します。板の表裏の広い面への放熱は含みません。その効果が重要な場合は 3D モデルなどで検討します。

34 実習：非定常熱解析で冷却を見る

最初は一つの簡単な直方体などを使い、全体の冷却を確認します。ここでの数値は操作例です。材料値と熱伝達率は実際の部品・流体・風速へ合わせてください。

1. ［解析の種類］で［非定常熱解析］を選び、形状を取り込みます。

2. 材料の熱伝導率・密度・比熱を確認し、メッシュを作成します。

3. 初期温度 100°C、解析終了時刻 60 秒、時間刻み 1 秒を設定します。

4. ［未設定境界を一括放熱］を True とし、周囲温度 20°C、熱伝達率 10 W/(m²K)を例として設定します。単に自然に冷える例では温度指定を加えません。

5. 解析を実行し、時刻スライダーで 0 秒から 60 秒へ進めて温度の変化を確認します。

設定	意味	変更による影響
初期温度	開始時の温度	冷却・加熱の出発点
温度指定	解析中ずっと維持する温度	指定箇所は開始時刻から指定値
解析終了時刻	物理現象を計算する長さ	何秒後まで求めるか
時間刻み	計算する時間の間隔	時間方向の精度と計算量
結果の保存数	記録する時刻の枚数の目安	表示・出力の密度とメモリー
再生（秒）	画面上の再生時間	計算済み物理時刻は変えない
計算の制限時間	処理の待ち時間の上限	解析終了時刻とは別

全体が断熱で加熱条件もなければ、初期温度を保ちます。時間刻みを小さくした計算と比較し、温度履歴の変化を確認します。材料値や条件は解析中一定で、時間表による切替、放射、相変化、温度依存物性は本版の対象外です。

35 変位・応力・反力を読み分ける

最初に見る三つの点

1. 変形の向きが荷重と支持の想定に合うか。変形倍率を大きくした表示では、実寸の変形と区別します。

2. 外力と反力の合計が釣り合うか。例えば外力－100 N なら、対応方向に合計＋100 N の反力が目安です。

3. 高い応力がどこに集中しているか。固定端や点荷重の直近だけでなく、評価したい範囲を確認します。

結果表示	読み方
変位	mm。変位量は大きさ。表示倍率で計算値は変わらない
ミーゼス応力	N/mm²。要素の相当応力で、許容値判定そのものではない
反力	拘束が物体へ加える力。全体合計と選択範囲合計を区別
垂直ひずみ	mm/mm。正が伸び、負が縮み
工学せん断ひずみ	対象平面のせん断。垂直ひずみと区別
相当弾性ひずみ	現行の定義ではミーゼス応力／E。塑性ひずみではない

色表示と最大値

滑らかな表示は節点間を補間し、応力・ひずみ等は表示用に平均します。見た目が滑らかでもメッシュ精度が増えるわけではありません。凡例は表示対象の値に基づくので、内部要素を含む数値と異なることがあります。必要な全要素の値は数値出力で確認します。

［元の輪郭］は変位表示で変形前を比較する機能です。応力の色だけで変位を判断しないでください。黄色の選択が邪魔な場合は［選択解除］または Esc を使います。条件や結果は消えません。

比較資料に残すもの

材料、形状版、メッシュ寸法と要素数、拘束、荷重の合計・方向、表示成分、単位、変形倍率、熱解析なら時刻を記録します。異なる図で凡例範囲が異なる場合、同じ赤色でも同じ値とは限りません。

36 条件・結果の保存と出力

解析画面の保存と図面ファイルの保存は二段階です。［条件を図面に保存／格納］を実行した後、通常の JDC 保存を行ってください。

保存・出力	内容	再利用
条件を図面に保存／格納	材料値、メッシュ／部材、境界条件など	JDC を開き、解析画面で再計算
結果を書き出す：JSON	条件と数値結果	記録・外部での集計用
節点反力 CSV	節点座標、方向別反力など	合計・範囲別の確認
要素ひずみ CSV	成分と相当弾性ひずみ	内部を含む要素結果の確認
非定常の温度 CSV	保存時刻別の節点温度	温度履歴の確認
画像保存・印刷	現在の視点、表示項目、凡例等	検討資料
梁の結果を CAD に配置	選択中の結果図、倍率や最大値	図面上の説明図

条件は 2D／3D・解析種類ごとに 1 組です。条件違いを比較する場合は、条件名の分かる別名 JDC と結果ファイルを用意します。定常／非定常の熱条件は 2D・3D それぞれで共有するため、別ケースとして残す場合もファイルを分けます。

保存済みの材料

解析条件には保存時点の材料値を保持します。共通の材質リストを後から変更しても、保存済み条件を自動で書き換えません。新しい材料値を使う場合は材料を選び直して再計算します。

出力前の画面操作

全体表示で対象を収め、必要な結果と倍率を選びます。画像保存では選択中の黄色表示や操作案内を除き、凡例等を含めます。非定常では保存する時刻を選びます。数値結果を書き出すことと、画像の再生表示を動画ファイルにすることは別で、動画保存は本版の対象外です。

37 解析で困ったときの確認表

症状	主な確認箇所	対処
条件を付ける面が選べない	メッシュ作成前か、選択方法	メッシュを作り、面／稜線／頂点を選ぶ
2D 取り込みに失敗	外周の閉鎖、同一平面、複数外周	一つの外周と穴に整理
3D メッシュが作れない	開いた形状、体積重複、微小形状	元の BRep と部品の位置を確認
拘束不足で計算できない	剛体移動・回転、独立部品	実際の支持・接合を設定
荷重が想定より大きい	重複行、分布と合計の混同	総荷重と反力を照合
変位が大きく違う	E、断面 I、板厚、単位、軸	mm・N 系と曲げ軸を確認
メッシュを細かくすると最大応力が上昇	角、点荷重、固定端	評価箇所と実際の接触範囲を見直す
全体が同じ温度	断熱、温度指定、定常を選択	冷却過程なら非定常を使う
非定常で冷えない	放熱の無効、周囲温度、温度固定	一括／個別放熱と初期温度を確認
CAD を直しても結果が変わらない	取り込み済み形状	再取り込み・再メッシュ・条件設定
保存後の値が古い	格納後に JDC を保存したか	条件格納と図面保存を両方行う
計算が長い	要素数、制限時間、モデル規模	目的に影響しない細部を整理して段階的に解析

不成立のモデルに対して、制限時間やメッシュ密度だけを増やしても原因は解消しません。形状→材料と単位→接合→拘束→荷重の順に確認すると、原因を絞りやすくなります。

第 5 部 IFC 部品と Revit ファミリ

形状・分類・属性・接続口を設定し、IFC から RFA へ変換する手順を説明します。

01 IFC から Revit ファミリまでの流れ.....354

IFC ライブラリ書き出しへの入口

［挿入］→［書き出し］→［IFC ライブラリ書き出し］を実行します。利用できるのは STD・Pro です。1 部品を構成するソリッドを選び、分類・属性・接続口を設定します。

IFC4 の曲面 BRep、複数ソリッドとブロックの扱い、Revit 2026／2027 用 Dynamo による RFA 作成は、第 5 部にまとめています。通常の CAD 書き出しと異なり、部品分類や設備の接続口を設定してから出力するコマンドです。

02 形状・基点・出力形式を選ぶ.....355
03 分類・標準属性・追加仕様.....356
04 接続口を 3D 形状から指定する.....357
05 給水・給湯・汚水・ガスの接続口.....358
06 電源 IN コネクタを設定する.....359
07 設定を保存して IFC を書き出す.....360
08 Dynamo Player から RFA を作る.....361
09 Revit で属性と接続口を確認する.....362
10 変換範囲とエラー時の対応.....363
分類一覧：58 分類の日本語と IFC4 対応.....364

01 IFC から Revit ファミリまでの流れ

IFC ライブラリ書き出しは、JDesignCAD で作った一つの部品へ分類・製品情報・接続口を付けて出力する機能です。IFC の出力は STD・Pro、RFA への変換は Revit 2026 または 2027 と付属の Dynamo を使います。

段階	使用する機能	作成・確認するもの
1	JDesignCAD の 3D 作図	部品のソリッド群、色、基点
2	IFC ライブラリ書き出し	分類・属性・接続口の設定
3	IFC2x3／IFC4 出力	1 部品の IFC ファイル
4	Revit の Dynamo Player	IFC 選択→RFA 保存
5	Revit	ファミリタイプ、材料、接続口、プロジェクトでの接続

一つの部品と複数ソリッド

天板・脚・幕板などを別のソリッドで作っても、一つの部品として書き出せます。構成ソリッドを一つの塊へブーリアン結合する必要はありません。一つの IFC に一つの部品実体とタイプを記録し、複数ソリッドをその形状として扱います。

「IFC ライブラリ」はこの部品出力の名称です。IFC 規格の全クラス、複数品番を一括管理する IfcProjectLibrary、Revit のパラメトリックな作成履歴を生成する機能ではありません。収録する日本語分類は 58 種類です。

ファミリの形状と属性

Dynamo で作る RFA はロード可能な固定形状のファミリです。メーカー・型番・仕様を保持し、対応条件の接続口を Revit ネイティブコネクタにします。呼称幅・高さの値を変えても、本体形状は伸縮しません。

家具等の静的な部品、設備の配置と対応する配管・電源接続に利用します。壁の開口、光源、構造解析モデルなどの専用機能は別途 Revit 側で作成が必要です。

02 形状・基点・出力形式を選ぶ

1. 出力する部品を構成するソリッドをすべて選びます。ソリッド群を含む対応ブロックも選択できます。

2. ［挿入］→［書き出し］→［IFC ライブラリ書き出し］を実行します。

3. ［基本情報・基点］で部品名、型番、メーカー等と配置の基点を設定します。

4. 出力形式を選びます。曲面を BRep として渡したい場合は［IFC4 曲面 BRep］を使います。

形式	形状表現	選択の目安
IFC4 曲面 BRep	IfcAdvancedBrep	曲面と稜線を持つ部品。標準の選択
IFC4 面分割 BRep	IfcFacetedBrep	受け取り側が面分割を必要とする場合
IFC2x3 面分割 BRep	IfcFacetedBrep	IFC2x3 を指定された受け渡し

曲面 BRep では、直接表せない曲面や一部の境界を、0.001 mm の精度検査を通した範囲付き NURBS へ変換する場合があります。すべての曲面を数学的に完全同一の形式へ変換するという意味ではありません。

変換に失敗したとき、黙って面分割へ切り替えたり面を欠落させたりしません。面分割を選ぶ場合は分割精度を設定し、閉鎖性・面の向き・体積を確認して出力します。

ダイナミックブロックを出す場合

出したい寸法・ルックアップ・可視状態へ変更してから選択します。現在の可視形状を評価して、複数ソリッドの部品として扱います。通常は事前の分解は不要です。ダイナミックな動作定義や非表示の別仕様を、Revit へ引き継ぐ出力ではありません。

受け取り側で配置しやすい基点を決めてください。Dynamo は IFC へ出した座標を保ち、自動で中心へ移す処理は行いません。

03 分類・標準属性・追加仕様

分類を選ぶと、日本語の項目と IFC 属性名の対応が表示されます。分類の案内に従って、実製品の情報を入力します。表示されたすべての欄が IFC 規格上必須という意味ではありません。受け渡し先が要求する項目も確認してください。

タブ	入力するもの	例
基本情報・基点	部品名、型番、メーカー、説明等	OA テーブル、型番 001
分類の標準属性	分類に対応する仕様	呼称高さ、呼称長さ、主な色
追加の製品仕様	標準欄にない名称と値・単位	天板材質：メラミン化粧板
接続口 IN / OUT	接続位置、方向、分野、系統等	給水 IN、電源 IN

家具の寸法と色の例

家具／テーブルの呼称高さ 720 mm、呼称長さ 1,800 mm、呼称奥行 800 mm を入力します。これは製品仕様としての値で、外形から自動取得した寸法ではありません。実際のモデルと一致する値を入力してください。

［主な色］には「ホワイト」などの文字を入力します。この文字でソリッドの表示色が変わるわけではありません。形状の色・透過率は CAD 側の面や材質の設定で確認します。

追加の製品仕様の例

仕様名は日本語で入力できます。「定格消費電力」に「100 W」、「取付方式」に「床置き」など、値の意味が分かる単位・表記を使います。Dynamo の電源仕様に使う 3 項目は指定名と数値だけを使うため、後の電源接続口の項に従ってください。

IFC2x3 で標準定義が異なる属性は、JDC_から始まる独自属性へ単位付きで補完し、完了時に通知します。相手ソフトで表示される場所や属性名が同じになるとは限りません。

04 接続口を 3D 形状から指定する

形状から位置・法線・開口寸法を取得し、必要な値だけ数値で補います。IN／OUT、系統、呼び径など製品の意味は形状だけでは決まらないため、必ず確認します。

1. ［接続口 IN / OUT］で［3D 形状から追加］を押します。既存の行を直す場合は［選択口を 3D 形状から設定］を使います。

2. 接続面をクリックします。この面が部品から外向きの方向の基準になります。

3. 開口の輪郭をクリックします。円なら中心と径、矩形なら中心・幅・高さ・断面回転を取得します。

4. 表に戻り、口名、分野、系統、流れ、呼び径・寸法を確認します。必要な数値を修正します。

5. 軸が逆なら［選択口の方向を反転］を使います。形状・基点との位置関係も確認します。

指定する形	取得内容・補足
円・円弧の開口	中心と内径／外径
同心円 2 輪郭だけの端面	内径・外径を同時取得
複数穴を持つフランジ面	選択した輪郭の寸法。目的の開口を選ぶ
直線からなる矩形	中心、幅、高さ、断面回転
その他の輪郭	［面上の点］と数値入力を組み合わせる

位置は WCS の mm、方向は部品から外向きのベクトルです。流れの IN／OUT とは別の意味です。IN を選ぶために、入力した外向き軸を利用者が内向きへ入れ替える必要はありません。

Esc で中止した場合は既存の設定を保持します。形状取得後も各値は数値修正できます。Revit のコネクタ変換では、対応する平らな端面上または円形開口中心に置く必要があります。

05 給水・給湯・汚水・ガスの接続口

IFC の接続口と、Dynamo で Revit コネクタへ変換できる範囲は異なります。同梱 Dynamo で配管コネクタへ変換する場合は次の条件に合わせます。

項目	設定
分野	PIPE
断面	CIRCULAR（円形）
呼び径	mm で入力
給水の系統	DOMESTICCOLDWATER
給湯の系統	DOMESTICHOTWATER
汚水の系統	SANITARY
ガスの系統	GAS。Revit では「その他の配管」
流れ	IN＝SINK、OUT＝SOURCE、双方向＝SOURCEANDSINK

二つの接続口を持つ部品の例

簡単な弁の形状を作り、両端の平面上に「給水 IN」と「給水 OUT」を置きます。分野 PIPE、同じ給水系統、円形、呼び径 25 mm を例として入力し、流れだけを IN／OUT に分けます。各軸はそれぞれの端面から外向きです。

Dynamo は同じ系統の 2 口を持つポンプ・弁の内部リンクを設定します。3 口以上の内部経路や、異なる系統間のつながりは推測しません。NOTDEFINED の流れは指定し直してください。

Revit での確認

RFA を開き、接続口の記号を選んで系統・流れ・位置を確認します。ファミリタイプの接続径はコネクタへ関連付けられますが、本体の穴径は固定です。プロジェクトへロードし、対応する配管を実際に接続して確認します。

ガスはプロジェクト側にも「その他の配管」として適切なシステムを用意します。配布時の実機サンプル確認は給水・ガス・電源を対象としています。給湯・汚水も変換対象ですが、利用条件で接続を確認してください。

06 電源 IN コネクタを設定する

同梱 Dynamo は、電気の電源 IN を Revit の「電力－平衡」コネクタとして作成します。電源出力、制御・通信、ダクト・電線管コネクタは対象外です。

接続口の項目	設定
口名	電源 IN など、区別できる名前
分野	CABLE
系統	ELECTRICAL
流れ	SINK（IN）
位置・軸	平らな端面上、部品から外向き

追加の製品仕様へ入力する 3 項目

仕様名（この表記で入力）	値の例	入力の規則
電源電圧（V）	100	数値だけ。V を値へ付けない
電源極数	1	数値だけ
電源皮相電力（VA）	100	数値だけ。W とは区別する

これらは操作確認用の例です。実製品の定格に合わせてください。電源口が複数ある場合は「口名 / 電源電圧（V)」のように口名を付け、3 項目を各口に設定します。

Revit へ変換した後

接続口の記号を選び、電圧・極数・皮相電力・系統を確認します。ファミリタイプにも口名付きのパラメーターが追加され、コネクタへ関連付けられます。プロジェクトへロード後に電力回路を作成します。

分電盤の配電システム、回路表、負荷分類、接続先盤は自動設定しません。コネクタを作成できることと、プロジェクトの電気設備設定が完成していることは別です。受け取り側の構成に合わせて設定してください。

07 設定を保存して IFC を書き出す

1. 分類、基本情報、属性、形状の基点、接続口の位置と向きを確認します。

2. 確認のチェックを付け、［IFC 書き出し］を実行して保存先を指定します。

3. 完了時の通知を確認します。IFC2x3 の属性補完などがあれば、受け渡し先へ伝えます。

4. ［設定を図面に保存］を使用し、通常の JDC 保存も行って、次回の原本を残します。

設定が保存される場所

部品設定は選択した先頭要素へ保存されます。複数ソリッドの部品では、次回も同じソリッド一式を選択してください。図面へ保存した設定は元に戻す／やり直しに対応します。IFC ファイルを出力することと、元の JDC を保存することは別です。

出力できない場合

内容	確認する点
ソリッドを認識できない	閉じた形状か、線や開いた面だけを選んでいないか
形状の検査に失敗	面の境界・向き・体積、過去のブーリアン形状
曲面 BRep の変換に失敗	対象形状を確認。形式変更は受け取り要件と相談
属性の入力エラー	数値と単位、選択式の項目、必須となる設定
接続位置が意図と違う	WCS、基点、外向き軸、単位

形状を変えた後は、保存済みの接続口も確認し直します。穴や端面を移動しても、製品の意味を持つ接続情報が常に自動で追従するとは考えず、再取得または数値修正で合わせます。

IFC の部品間接続は未接続として出力します。部品内の口の定義と、建物全体でどの配管・機器につなぐかは別の設定です。

08 Dynamo Player から RFA を作る

インストール先の Revit/Dynamo、または単独配布 ZIP 内のフォルダーに、変換ファイルと「使い方.txt」があります。部品ごとに Dynamo ファイルを作り直す必要はありません。

使用する Revit	実行ファイル
Revit 2026	JDesignCAD_IFC からファミリ_Revit2026.dyn
Revit 2027	JDesignCAD_IFC からファミリ_Revit2027.dyn

初回の準備

1. 必要なら配布フォルダーを利用者の作業フォルダーへコピーします。

2. Revit でプロジェクトを開きます。空のプロジェクトでも構いません。

3. ［管理］→［Dynamo Player］を開き、.dyn を置いたフォルダーを登録・選択します。

4. ファイルの信頼確認が表示された場合は、入手元を確認して許可します。

毎回の操作

1. 使用中の Revit と同じ年版の変換を実行します。

2. ［JDesignCAD の IFC 部品を選択］で出力した IFC を選びます。

3. ［ファミリを保存］で RFA の名前と保存先を指定します。

4. 完了後、保存した RFA を開くかプロジェクトへロードします。

ファミリテンプレートを自動で見つけられない場合だけ、.rft を選択します。ホストなしの「一般モデル（メートル単位)」または家具テンプレートを使います。壁・天井などのホストや既存形状を含むテンプレートは使用しません。

追加の製品アドインや Dynamo パッケージは不要です。Revit LT と Dynamo 単体は対象外です。IFC リンク、バインド、グループ解除を先に行う必要はありません。

2026 版は CPython3、2027 版は PythonNet3 用です。異なる年版では中止します。2027 で保存した RFA は 2026 で開けません。両方へ配布する場合は 2026 で作成し、2027 でアップグレードして利用する方法を選びます。

09 Revit で属性と接続口を確認する

ファミリエディタで見る

RFA を開き、［ファミリタイプ］を開きます。部品名、IFC 分類、メーカー、型番、標準属性と追加仕様を確認できます。メーカーは標準の［製造元］、型番は［モデル］にも設定します。

プロジェクトへロードした場合は、配置した部品を選んで［タイプ編集］を開きます。個別の配置情報はプロパティパレット、共通仕様はタイプの設定を確認します。

項目	引き継ぎ・注意点
IFC プロパティセット	元の属性名を持つパラメーターを保持
標準属性	日本語の項目も追加
長さ	Revit の長さ型
真偽値	Yes/No 型
その他の値	単位を含む文字列
面の色・透過率	Revit が読み取ったシェーディング用材質
テクスチャ	自動作成しない

接続口を選ぶ

RFA 内で接続口の記号を選び、左のプロパティで系統・流れ・説明を確認します。［ファミリタイプ］の「給水 IN 接続径」「電源 IN 電圧」などはコネクタへ関連付けられます。

接続径を変えても、固定形状の穴径や外形は変わりません。また、元 IFC の仕様属性は取り込み時点の記録で、コネクタ値の変更へ自動追従しません。納品時には本体、コネクタ、仕様が矛盾しないことを確認します。

IFC リンクで見ている場合

リンク全体のタイププロパティやプロジェクト情報は、部品のファミリタイプとは異なります。ロード可能ファミリとして使う目的なら、本章の Dynamo で RFA を作成し、その RFA を開いて確認してください。

10 変換範囲とエラー時の対応

家具、衛生器具、機械設備などは対応する Revit カテゴリへ設定します。壁・床・屋根・直管・直線ダクト・梁など、変換器が専用カテゴリへ対応していない分類は、一般モデルの固定形状として保存し、その旨を表示します。IFC 分類は属性に残します。

対象	同梱 Dynamo の範囲
部品形状	Revit がソリッドとして読める形状。複数ソリッド対応
部品仕様	基本情報、標準属性、追加仕様を転送
配管口	円形の給水・給湯・汚水・ガス
電源口	CABLE／ELECTRICAL／SINK の電力－平衡
その他の接続口	未対応の口がある場合は中止して通知
ドア・窓	壁を切り欠くホスト機能は作成しない
光源・構造解析モデル	作成しない
対象 IFC	JDesignCAD が出力した 1 部品。建物全体の汎用変換ではない

保存前後の確認

形状数・体積・位置・属性、接続口の個数・位置・方向・系統などを検査します。ソリッド不足、メッシュ化、色の対応不明、属性転送失敗などでは処理を中止します。一部を黙って省略した RFA を完成品として保存しません。

元の IFC は一時フォルダーへコピーして処理します。開いているプロジェクトへリンクや図形を追加しません。検査済みの一時 RFA を指定先へ移すため、通常の変換失敗やキャンセルでは既存 RFA を変更しません。ただし RFA ファイルの保存は、プロジェクトの UNDO 対象ではありません。

問題が出た場合

エラーの［詳細］、Revit の年版、使用した.dyn、対象 IFC、目的の接続口名を確認します。接続口では平らな端面上にあるか、軸・系統・流れが明確かを先に確認してください。対応しない口を外して形状だけを作る場合は、接続機能がない部品として扱います。

第 5 部 分類一覧の参照

IFC4 での分類を確認するための一覧です。IFC2x3 では対応する実体・タイプへ変換され、画面に対応名が表示されます。

分類一覧 01 日本語分類と IFC4 の対応 365
分類一覧 02 日本語分類と IFC4 の対応 366
分類一覧 03 日本語分類と IFC4 の対応 367
分類一覧 04 日本語分類と IFC4 の対応 368
分類一覧 05 日本語分類と IFC4 の対応 369
分類一覧 06 日本語分類と IFC4 の対応 370

分類一覧 01 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
衛生／便器	IfcSanitaryTerminal / TOILETPAN	給水 IN、汚水 OUT。温水洗浄便座などの電源は条件付
衛生／小便器	IfcSanitaryTerminal / URINAL	給水 IN、汚水 OUT。自動洗浄用電源は条件付
衛生／洗面器	IfcSanitaryTerminal / WASHHANDBASIN	排水 OUT。水栓を含む部品なら給水・給湯 IN
衛生／流し台のシンク	IfcSanitaryTerminal / SINK	排水 OUT。水栓を含む場合は給水・給湯 IN
衛生／浴槽	IfcSanitaryTerminal / BATH	給湯 IN、排水 OUT。追いだき循環等は製品に応じ追加
衛生／シャワー	IfcSanitaryTerminal / SHOWER	給水・給湯 IN。混合／吐出口は実製品の構成による
衛生／洗浄水槽	IfcSanitaryTerminal / CISTERN	給水 IN、便器への洗浄水 OUT
衛生／便座	IfcSanitaryTerminal / WCSEAT	普通便座は不要。温水洗浄なら給水 IN・電源等
配管設備／循環ポンプ	IfcPump / CIRCULATOR	吸込 IN、吐出 OUT、電源。ドレン等は必要時
配管設備／片吸込ポンプ	IfcPump / ENDSUCTION	吸込 IN、吐出 OUT、電源

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

分類一覧 02 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
配管設備／水中ポンプ	IfcPump / SUBMERSIBLEPUMP	吐出 OUT、電源。自由吸込を配管接続口と混同しない
配管設備／直管	IfcPipeSegment / RIGIDSEGMENT	通常 2 口。固定方向のない管は双方向／未定も選択可
配管設備／可とう管	IfcPipeSegment / FLEXIBLESEGMENT	通常 2 口。可とう部の姿勢と端部の向きを分けて保持
配管設備／エルボ	IfcPipeFitting / BEND	2 口。位置・軸・接続方式を口ごとに設定
配管設備／チーズ・分岐継手	IfcPipeFitting / JUNCTION	通常 3 口以上。流向は実際の系統で設定
配管設備／異径継手	IfcPipeFitting / TRANSITION	2 口。異なる径と偏心量を保持
配管設備／フランジ継手	IfcPipeFitting / CONNECTOR	配管接続面を設定。ボルト穴は接続口にしない
配管設備／止め弁	IfcValve / ISOLATING	通常 2 口。ボール／ゲート等は構造属性で設定
配管設備／逆止弁	IfcValve / CHECK	入口 IN、出口 OUT。逆方向を許容しない用途を確認
配管設備／減圧弁	IfcValve / PRESSUREREDUCING	高圧側 IN、低圧側 OUT

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

分類一覧 03 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
配管設備／水栓	IfcValve / FAUCET	給水／給湯 IN、吐水側 OUT。実配管口と自由吐水を区別
配管設備／給水貯水槽	IfcTank / STORAGE	流入 IN、流出 OUT。排水・越流・通気等は必要時
配管設備／ボイラー（水用）	IfcBoiler / WATER	水の入口／出口。燃料・排気・電源は機種に応じ追加
空調／空調機（AHU）	IfcUnitaryEquipment / AIRHANDLER	給気・還気・外気・排気。熱源水やドレンは条件付
空調／パッケージ空調機	IfcUnitaryEquipment / AIRCONDITIONINGUNIT	冷媒管、ドレン、電源。ダクト接続は機種による
空調／軸流ファン	IfcFan / PROPELLORAXIAL	吸込 IN、吐出 OUT、電源。開放端の有無を区別
空調／直線ダクト	IfcDuctSegment / RIGIDSEGMENT	通常 2 口。矩形は幅・高さ・断面の回転も保持
空調／ダクト曲がり	IfcDuctFitting / BEND	通常 2 口。口ごとに中心・方向・断面を設定
空調／ダンパー	IfcDamper / CONTROLDAMPER	通常 2 口。電動操作の電源・制御は条件付
空調／吹出口（ディフューザー）	IfcAirTerminal / DIFFUSER	ダクト接続口。吹出／吸込の用途は系統で確認

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

分類一覧 04 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
空調／グリル	IfcAirTerminal / GRILLE	接続口を設定。見た目で給気／還気を決めない
消防／スプリンクラーヘッド	IfcFireSuppressionTerminal / SPRINKLER	配管接続 IN。放水先を架空の配管口にしない
電気／配電盤・分電盤	IfcElectricDistributionBoard / DISTRIBUTIONBOARD	受電・出力・制御を必要に応じ設定。各口の役割を区別
電気／コンセント（電源）	IfcOutlet / POWEROUTLET	電気接続。回路上の接続とプラグの挿し口を区別
電気／照明器具（点光源型）	IfcLightFixture / POINTSOURCE	電源。調光・制御線は条件付
電気／ケーブルトレイ	IfcCableCarrierSegment / CABLETRAYSEGMENT	トレイの機械的接続口。電力の IN/OUT を一律設定しない
電気／電線管	IfcCableCarrierSegment / CONDUITSEGMENT	通常 2 口。管路の接続と電気回路を区別
家具／椅子	IfcFurniture / CHAIR	通常なし。電動機構等を持つ製品だけ追加
家具／テーブル	IfcFurniture / TABLE	通常なし。組込み電源などは条件付
家具／机	IfcFurniture / DESK	通常なし。組込みコンセント等は別設定

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

分類一覧 05 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
家具／ベッド	IfcFurniture / BED	通常なし。電動・医療用の電源等は製品別
家具／書類キャビネット	IfcFurniture / FILECABINET	通常なし
家具／棚	IfcFurniture / SHELF	通常なし
家具／ソファ	IfcFurniture / SOFA	通常なし。電動機構等は条件付
家具／その他の家具	IfcFurniture / USERDEFINED	製品用途による。USERDEFINED の説明名を必ず記録
構造／梁	IfcBeam / BEAM	設備ポートなし。構造接合や解析端点は別
構造／柱	IfcColumn / COLUMN	設備ポートなし。構造柱用途は LoadBearing を確認
構造／ブレース	IfcMember / BRACE	設備ポートなし。端部接合は構造の関係
構造／接合用鋼板	IfcPlate / SHEET	設備ポートなし。穴を配管口と解釈しない
構造／ボルト	IfcMechanicalFastener / BOLT	設備ポートなし。機械的な締結情報

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

分類一覧 06 日本語分類と IFC4 の対応

書き出し画面に収録した 58 分類の一覧です。IFC4 の要素名と事前定義種別を示します。製品ごとの標準属性は選択した分類の画面で確認し、追加仕様と接続情報を補います。

分類	IFC4 要素／種別	仕様・接続の確認例
構造／独立基礎	IfcFooting / PAD_FOOTING	設備ポートなし。支持部材との関係は別
建築／壁	IfcWall / STANDARD	通常は設備ポートなし。窓・ドアの開口／接合は別の関係
建築／床スラブ	IfcSlab / FLOOR	通常なし。開口や支持部材との関係は別
建築／屋根スラブ	IfcSlab / ROOF	通常なし。屋根全体と構成スラブを区別
建築／ドア	IfcDoor / DOOR	設備ポートは通常なし。自動扉の電源・制御は条件付
建築／窓	IfcWindow / WINDOW	通常なし。電動開閉時の電源・制御は条件付
建築／手すり	IfcRailing / HANDRAIL	通常なし。支柱や取付部の構成を保持
その他／未分類の建築部品	IfcBuildingElementProxy / USERDEFINED	必要な場合のみ。適切な専用 IFC 分類を先に探す

この表は IFC 書き出しの分類です。Revit コネクタの対応表ではありません。ダクト・制御通信等を IFC で設定できても、同梱 Dynamo が Revit ネイティブコネクタへ変換する範囲は本部 05・06・10 に限られます。

第 6 部 インストールと更新

配布された最新のインストーラー、または「更新を確認」から更新します。更新前の図面と部品原本を保存してください。

インストーラーから更新する

1. 作業中の図面を保存して、JDesignCAD を終了します。
2. 配布された「JDesignCAD-1.1.4-Setup.exe」を実行します。
3. インストーラーの案内に従って完了し、起動後のタイトルが「JDesignCAD Ver1.1.4」であることを確認します。

ZIP 版を使う

ZIP 版はインストーラー一式です。ZIP をすべて解凍し、解凍先の Setup.exe を実行します。ZIP 内から直接実行しないでください。EXE と MSI の両方を重ねてインストールする必要はありません。インストール不要のポータブル版ではありません。

自動更新を使えない環境

［ファイル］画面の［更新を確認］で案内を確認できます。自動更新を利用できない場合も、配布された新しいインストーラーから更新できます。

編集用データの保存

ダイナミックブロックの編集設定や解析条件を残す原本は JDC 形式で保存してください。他の CAD へ渡す DWG・DXF などは別名で保存し、必要な動作と形状が保持されていることを確認します。

本書は JDesignCAD 1.1.4 の操作をまとめています。画面例・解析例は操作説明用です。解析の仮定と交換形式の制限は各章を参照してください。

01 操作別の参照先

分野	操作内容	本書の参照先
表示言語	日本語／英語、起動パネル・主なダイアログ	第 1 部 25
モデル単位	mm・cm・m・km・inch・feet、実寸を保持	第 1 部 26〜28
データ交換	SXF、読み込み単位、STEP 改善	第 2 部 SXF の読み書き、第 1 部 28
クロソイド	通常曲線、IP 基本型、横断線等	第 2 部 クロソイド
モデリング	複数面編集、穴再編集、切断等の改善	第 2 部 面編集・穴・切断
部品・画層	画層 0 の継承、UCS 操作	第 1 部 レイヤ・座標系
ダイナミックブロック	3D 面動作、複数ソリッド、数式、グリップ	第 3 部 24〜33
解析	個別材料と完全接合、詳細な条件・結果の説明	第 4 部 25〜37
IFC と Revit	IFC4 曲面 BRep、属性・口、Dynamo 2026／2027	第 5 部
ファイルプレビュー	面の透過設定を反映	起動時パネル・プレビュー

本書の対象

基本操作と各機能の説明を本書に統合しています。別の版の追補を読み合わせる必要はありません。

解析の条件例、図解の数値、部品の寸法は学習用です。実際の用途では製品仕様と取付・荷重・材料・伝熱条件に合わせて設定します。未対応の専用機能や交換形式の制限は、各章に記載しています。

02 配布ファイルと原本の管理

配布ファイルの構成

ファイル・場所	用途
JDesignCAD-1.1.4-Setup.exe	JDesignCAD 本体のインストール／更新
インストーラー一式の ZIP	すべて展開して Setup.exe を実行
インストール先 Revit/Dynamo	2026／2027 用.dyn と使い方
JDesignCAD-IFC-RevitDynamo-1.1.4.zip	Revit 変換ファイルの独立配布
JDC	編集可能な図面・部品・解析条件の原本
IFC	分類・属性・接続口を持つ部品交換データ
RFA	実行した Revit 年版の固定形状ファミリ

作業データの残し方

ダイナミックブロックと 3D 面動作は JDC を原本とします。交換形式で保存したとき、動作や履歴をすべて保持するとは限りません。IFC・RFA を修正したい場合も、形状・接続口の元設定を持つ JDC を残しておくと更新しやすくなります。

解析はケースごとに JDC と結果出力を対応させ、形状版、材料、拘束・荷重、メッシュ、時刻を記録します。画像だけでは再計算に必要な条件が不足します。

更新後の確認

JDesignCAD を再起動して版番号を確認します。表示言語を変えた場合も再起動します。Dynamo は使用する Revit と同じ年版を選び、旧ファイルと取り違えないようフォルダーをそろえます。

旧版で失われた形状や、すでに線・文字へ分解した交換データは、ソフト更新だけで元の編集情報へ戻りません。更新前の保存ファイルとバックアップを保持してください。

第 7 部 板金コマンドリファレンス

板金は STD／Pro で利用します。板厚と曲げ条件を設定し、基本板または認識したソリッドを加工します。入力中に重い計算は行わず、［適用］で計算・検査し、成功した結果を 1 回の Undo で反映します。

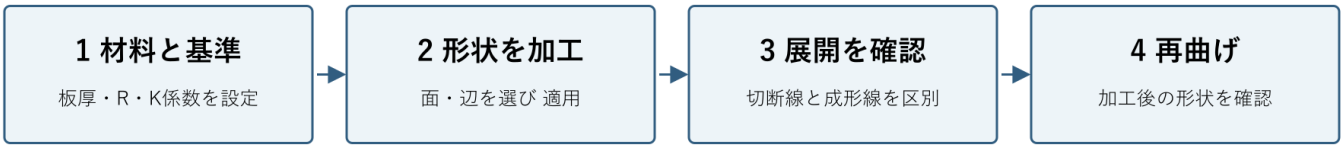


図 8　板金操作の基本的な順序。

成形の「展開」は基準平板への復帰と展開図表現です。絞り加工の材料流動・肉薄化・割れや製造用ブランク寸法を計算する機能ではありません。ルーバー／ブリッジの成形線と切断線を分けて扱ってください。

板金ダイアログ一覧

板金コマンドは、入力を確認して［適用］した時に計算します。成功した結果を 1 回の［元に戻す］で取り消せます。作成後は JDC で保存し、板厚・曲げ・展開情報を保持します。

基本板

閉じた輪郭を 1 つ選び、板厚 t と厚みの方向を指定して［適用］します。
XY 作業平面では、正方向＝上（＋Z)、負方向＝下（－Z）です。
両側均等は片側 t/2、合計 t。元の輪郭を残して板金ソリッドを作ります。

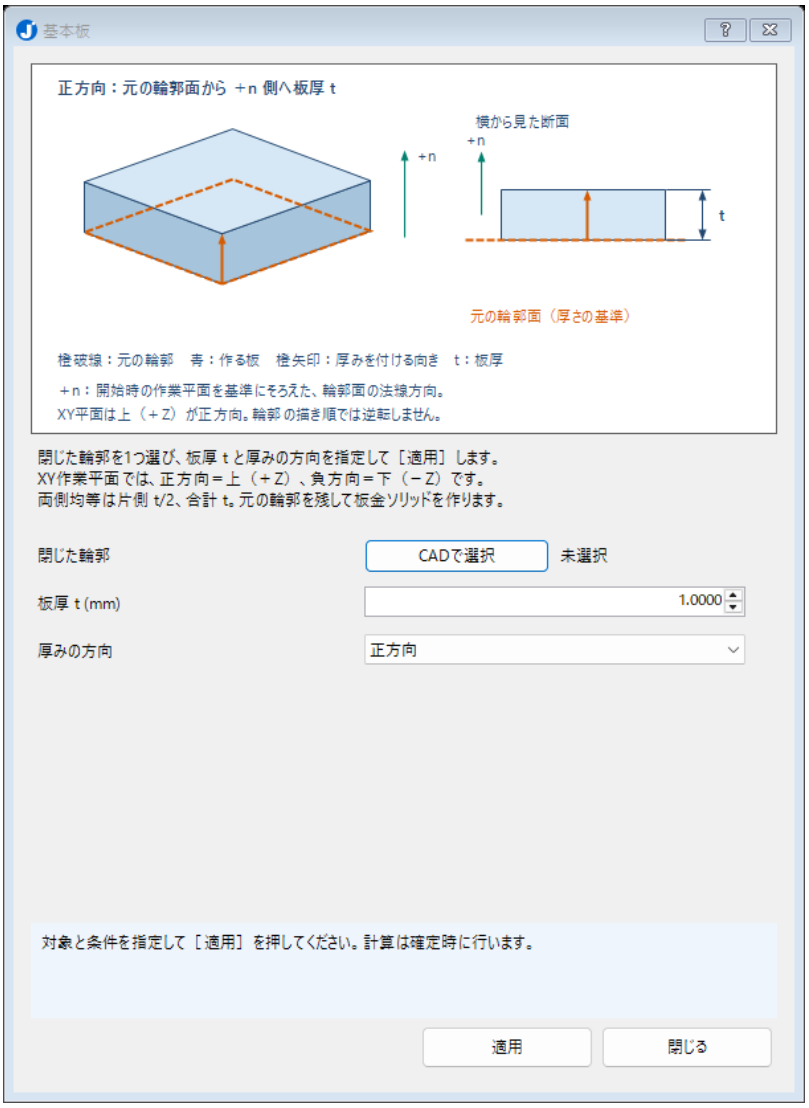


図 9　現在の操作画面。対象・入力値を設定して［適用］します。

線で曲げる

ソリッド→固定面→曲げ線の順に選びます。線の始点→終点の右側を曲げます。

既存の曲げを保持し、移動側に付いたフランジも一緒に動きます。

曲げ帯内の穴・既存の R 面や、曲げ後に干渉する形状は対象外です。

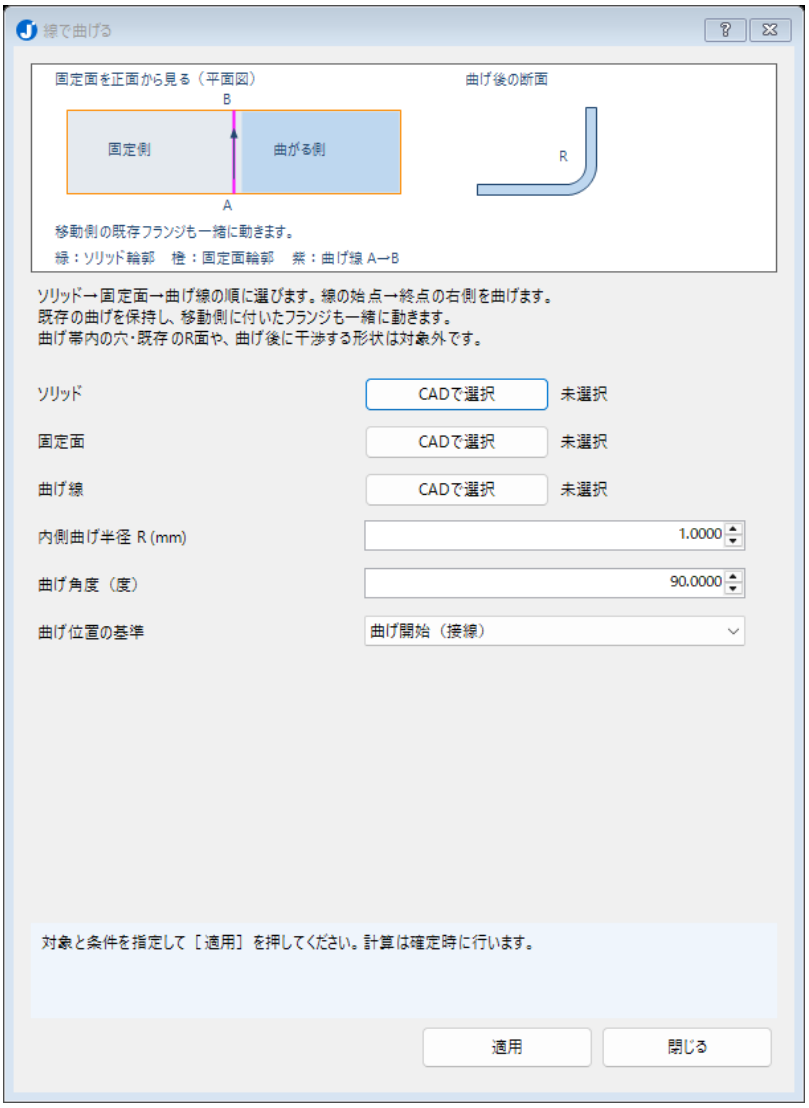


図 10　現在の操作画面。対象・入力値を設定して［適用］します。

曲げ管理

B 番号を指定し、抑制・再開・削除・親変更を行います。削除は子のフランジと成形も削除します。親変更は材料がつながる構成のみ有効です。計算と検査は［適用］時に行います。

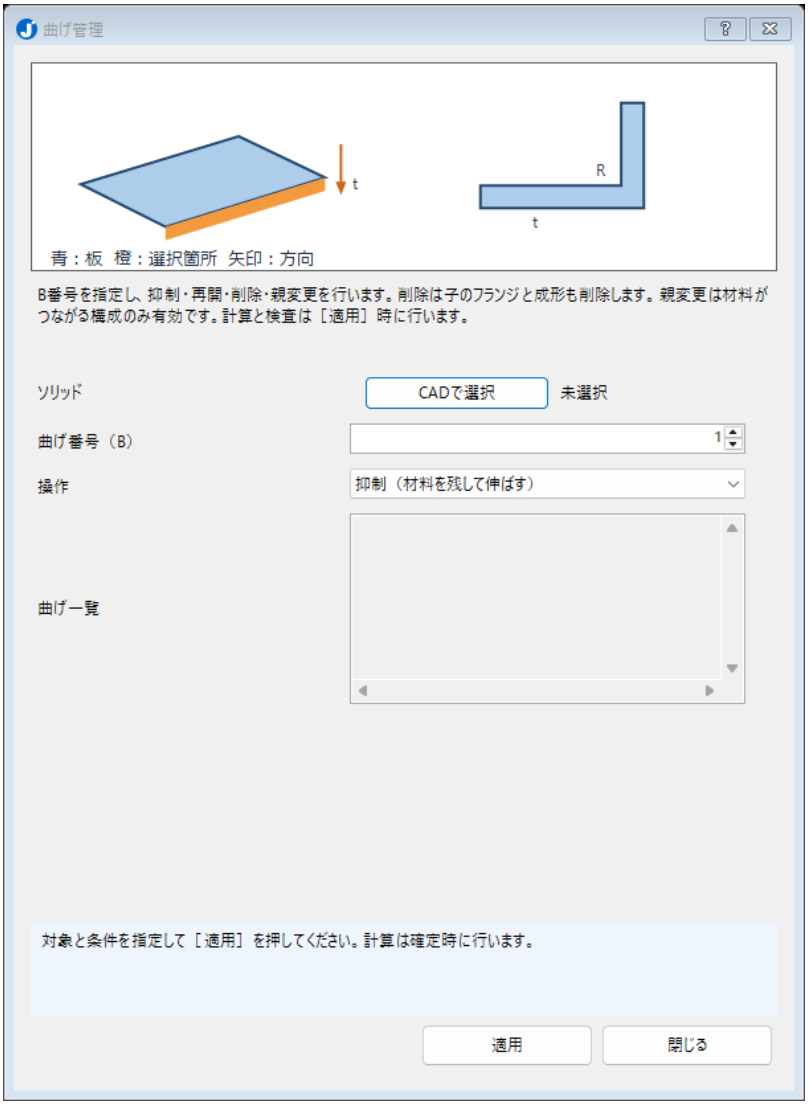


図 11　現在の操作画面。対象・入力値を設定して［適用］します。

曲げ逃げ

曲げ端部の円弧エッジを選び、幅 w・深さ d・形状を指定して［適用］します。

新規作成、または登録済みの逃げを変更します。［逃げなし］でその端の逃げを削除します。

既存の逃げ寸法は自動表示されません。変更したい値を入力してください。詳しくは［？］。

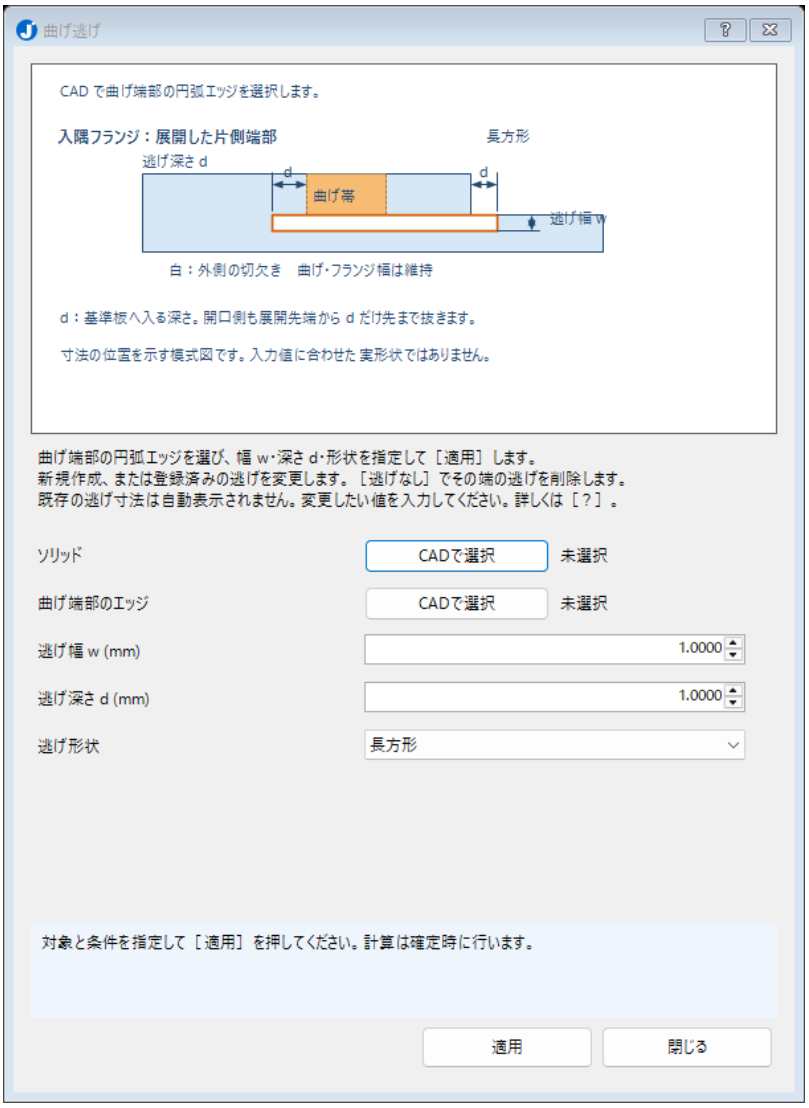


図 12　現在の操作画面。対象・入力値を設定して［適用］します。

展開補正表

Allowance は曲げ部分の展開長さ BA（mm）です。外寸合計から引く「伸び値」とは異なります。
対象と操作を選び［適用］で CSV を指定します。「選択した板金」は展開輪郭を保って曲げ代を更新します。書式・換算方法は下のリンクへ。

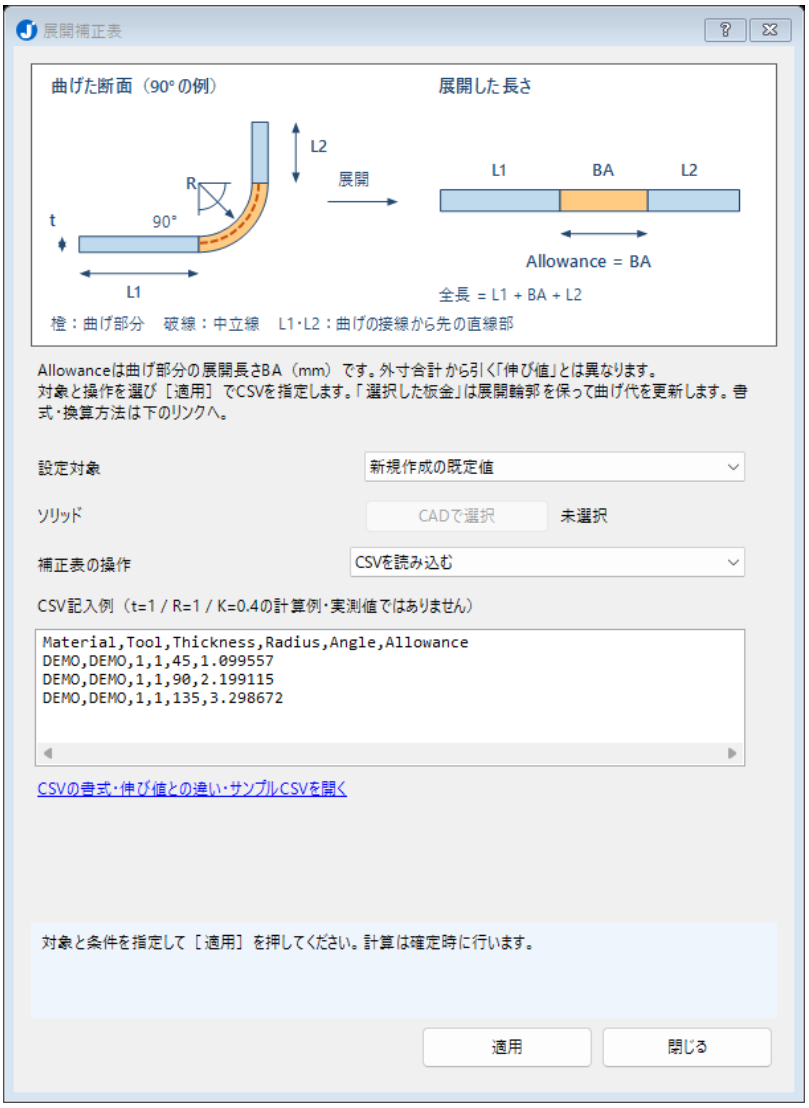


図 13　現在の操作画面。対象・入力値を設定して［適用］します。

板金チェック

一定板厚、閉じたソリッド、曲げの接続、曲げ／展開時の材料の重なりを検査します。

この検査は製造条件や曲げ機械の干渉を保証するものではありません。

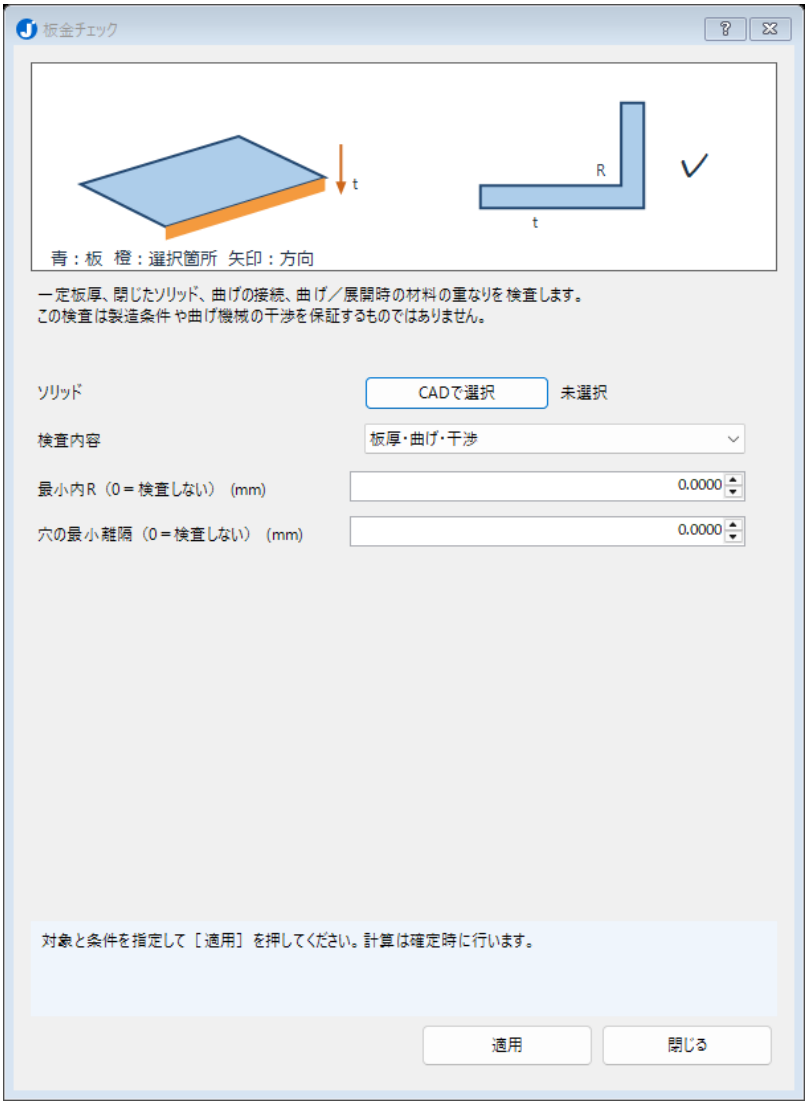


図 14　現在の操作画面。対象・入力値を設定して［適用］します。

連続フランジ

直線エッジを 1 本選びます。行ごとに直前面からの相対角度（負は逆向き）、曲げ後の直線部長さ、内 R を入力します。空行に追加でき、行頭を選んで Delete で削除します。長さは仮想交点間の寸法ではありません。

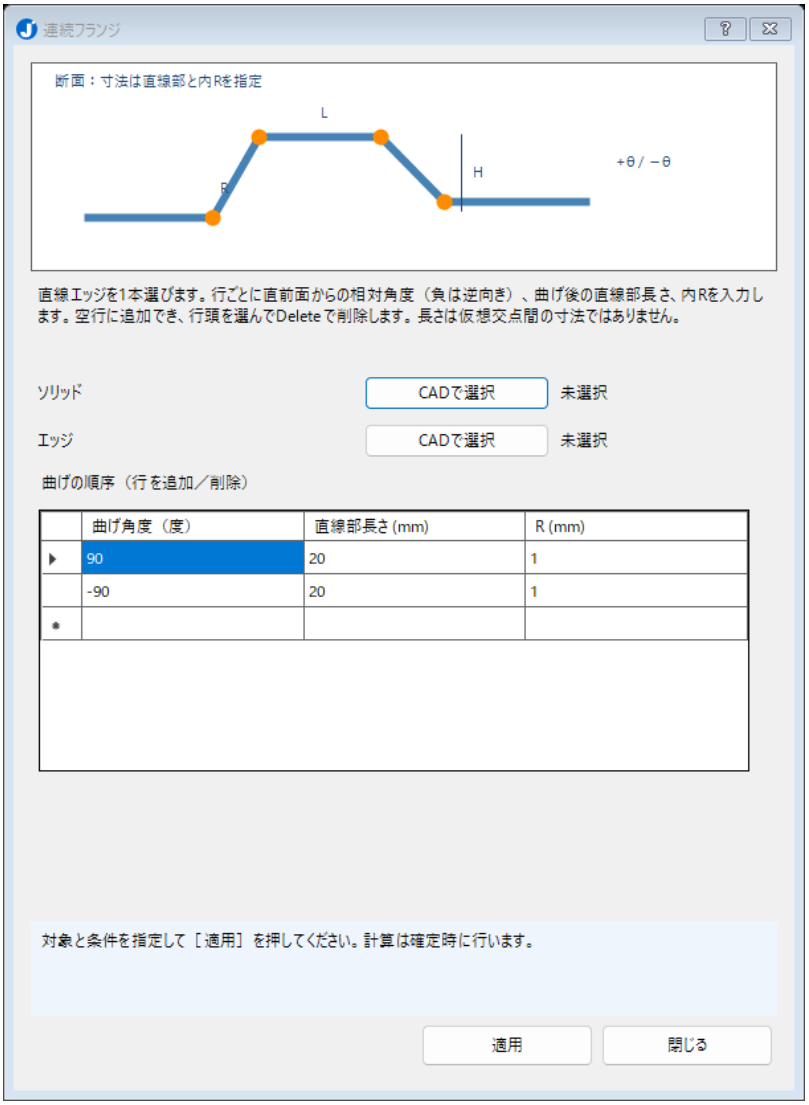


図 15　現在の操作画面。対象・入力値を設定して［適用］します。

コーナー処理

「45 度合わせ」では、同じ平面にある隣接フランジの広い面を 2 枚以上選びます。隙間を指定して［適用］します。

内向き・外向きの両方に対応します。加工前のフランジが対象です。「突合せ」は従来の曲げ端の退避です。

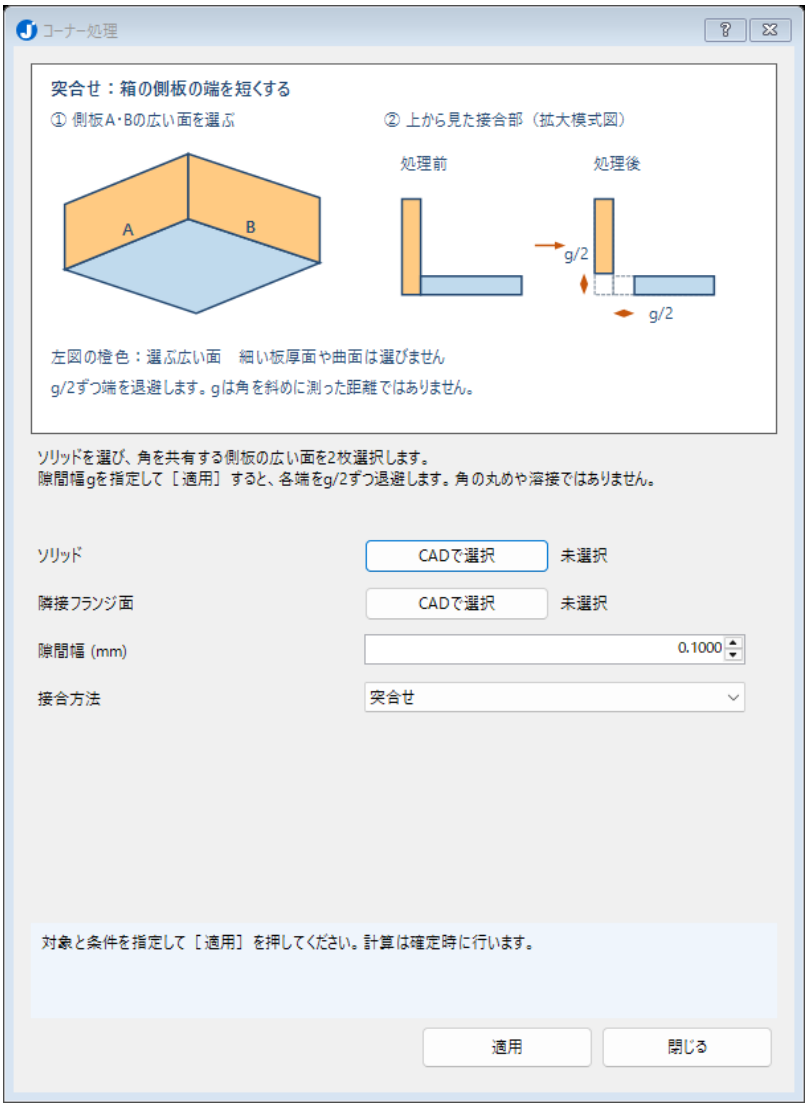


図 16　現在の操作画面。対象・入力値を設定して［適用］します。

エッジフランジ

直線エッジを選び、曲げ位置と長さの基準を指定します。従来と同じ寸法は「曲げ開始」「曲げ先の直線部」です。

箱の上部は「コーナー自動調整：自動で合わせる」で、隣接するフランジを 45 度に合わせます。隙間は合わせ線に直角の距離です。

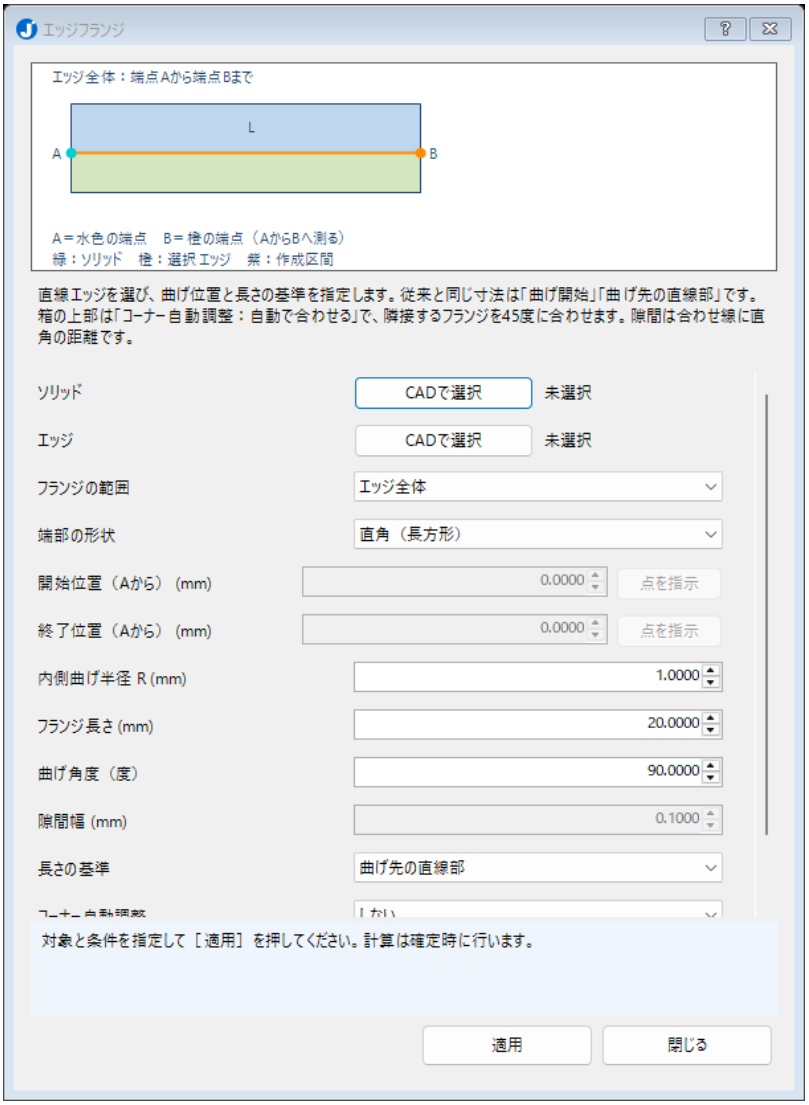


図 17　現在の操作画面。対象・入力値を設定して［適用］します。

曲げ・フランジ編集

ソリッドと円筒状の曲げ面を選び、角度・内側半径 R を変更します。

加工後も曲げをまたぐ穴・長穴・切欠きを保持します。加工済み／線曲げは展開輪郭を保つため、直線部の長さは自動計算します。

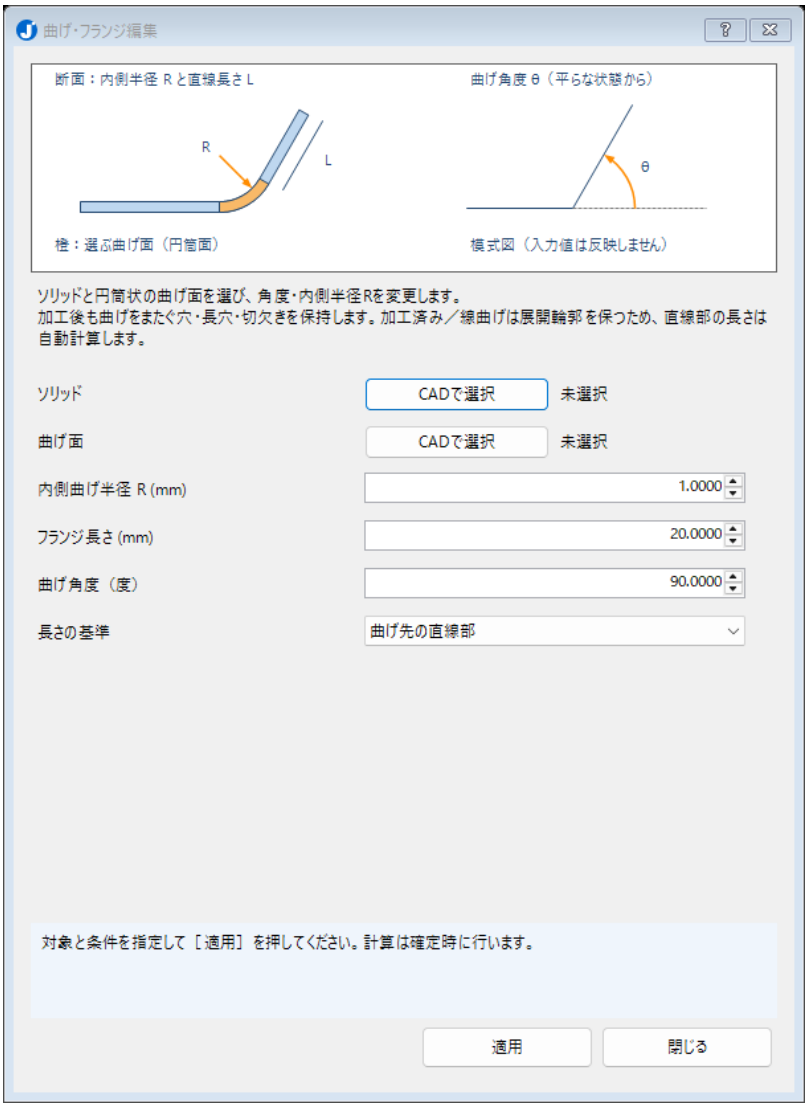


図 18　現在の操作画面。対象・入力値を設定して［適用］します。

展開データ出力

展開した外形・穴・曲げ中心線を DXF／DWG へ保存します。
出力単位を指定できます。元の図面とソリッドは変更しません。

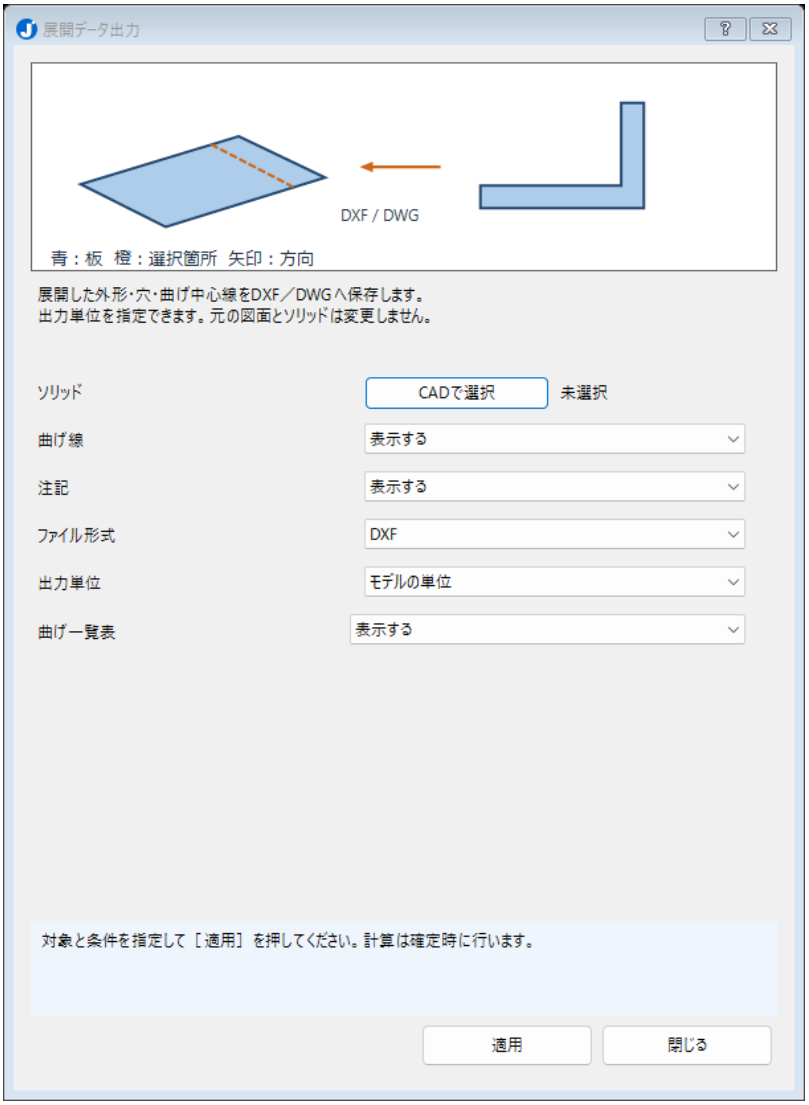


図 19　現在の操作画面。対象・入力値を設定して［適用］します。

展開図作成・更新

外形・穴・曲げ中心線・注記をモデル空間の XY 平面に配置します。

同じ部品で再実行すると前回の展開図を置き換えます。元のソリッドは残ります。

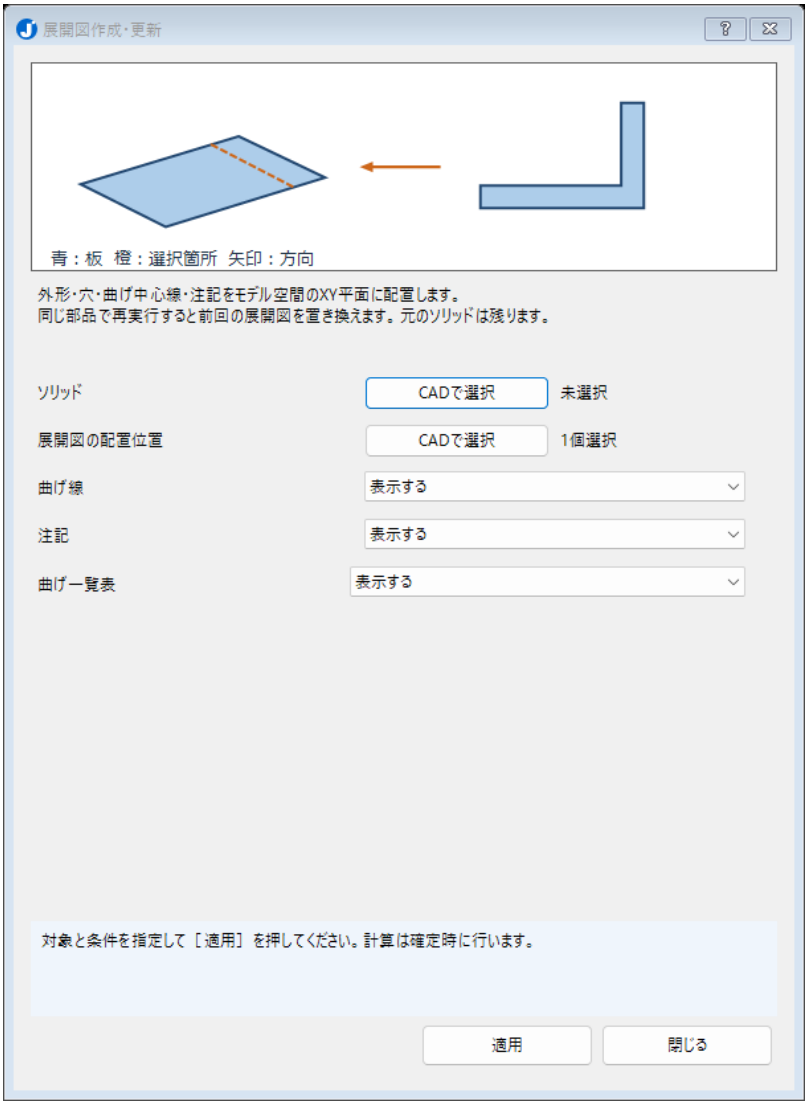


図 20　現在の操作画面。対象・入力値を設定して［適用］します。

板金成形（バーリング・ルーバー・絞り・ブリッジ）

ソリッド、平面、中心位置を選び、寸法を指定して［適用］します。展開時はバーリングの指定下穴を残し、ルーバーは平板と展開図の切込み線へ戻します。成形番号は作成順の F 番号です。肩 R は板厚より大きく、高さは根元 R＋肩 R 以上にします。

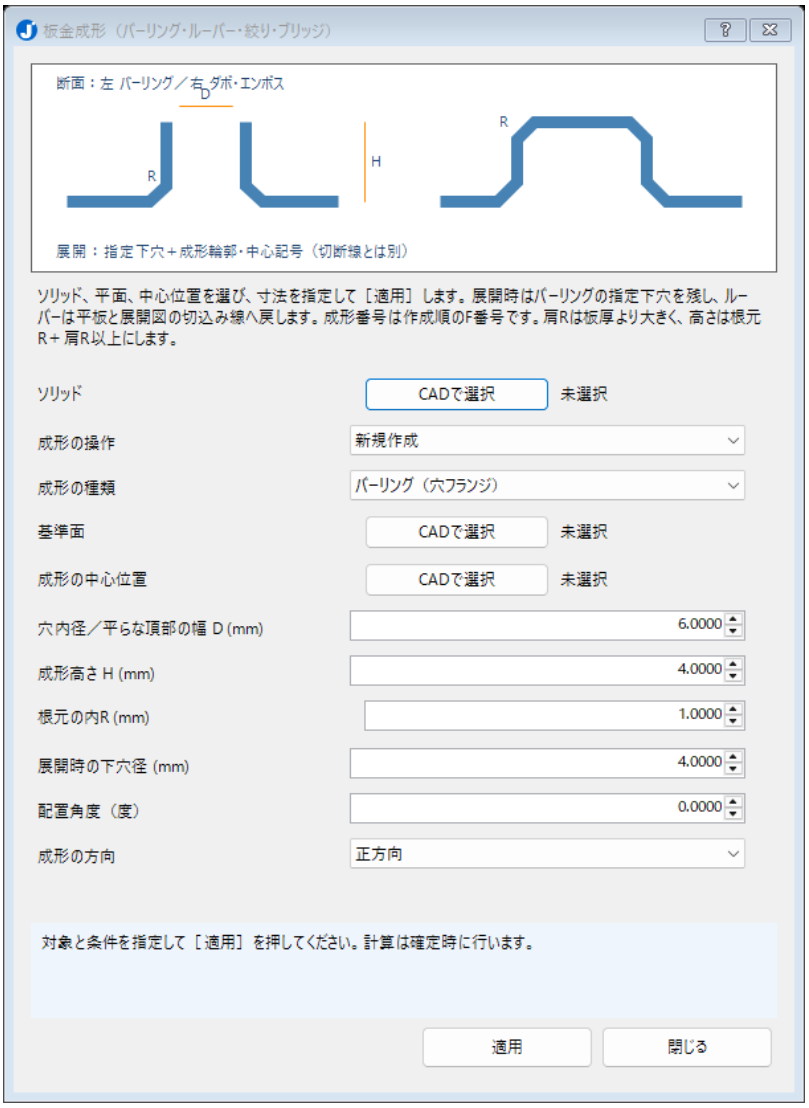


図 21　現在の操作画面。対象・入力値を設定して［適用］します。

折返し（開きヘム）

直線エッジを選び、折返しの直線長さと板間の隙間を指定します。内 R＝隙間／2 の 180 度の開きヘムです。隙間ゼロのつぶしヘムではありません。

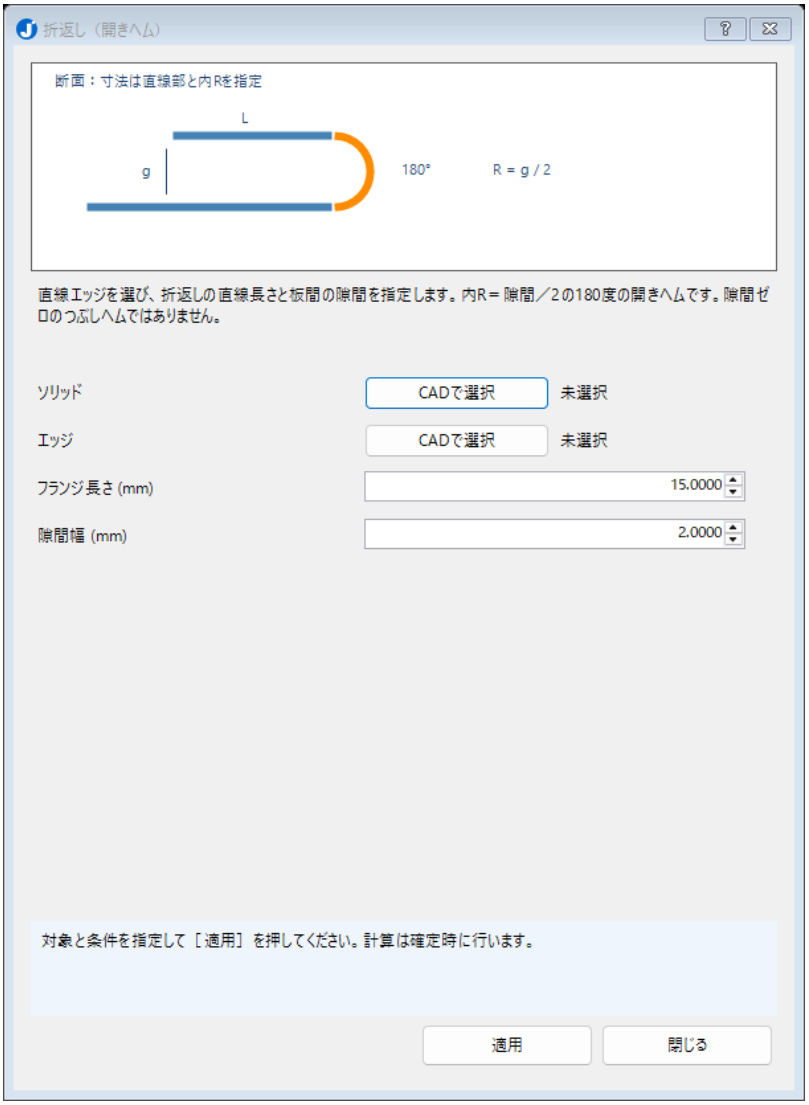


図 22　現在の操作画面。対象・入力値を設定して［適用］します。

段曲げ

直線エッジを選び、段差高さ・先端直線長さ・内 R・角度を指定します。90 度では高さ＞2R＋板厚が必要です。二つ目は逆方向に曲げます。

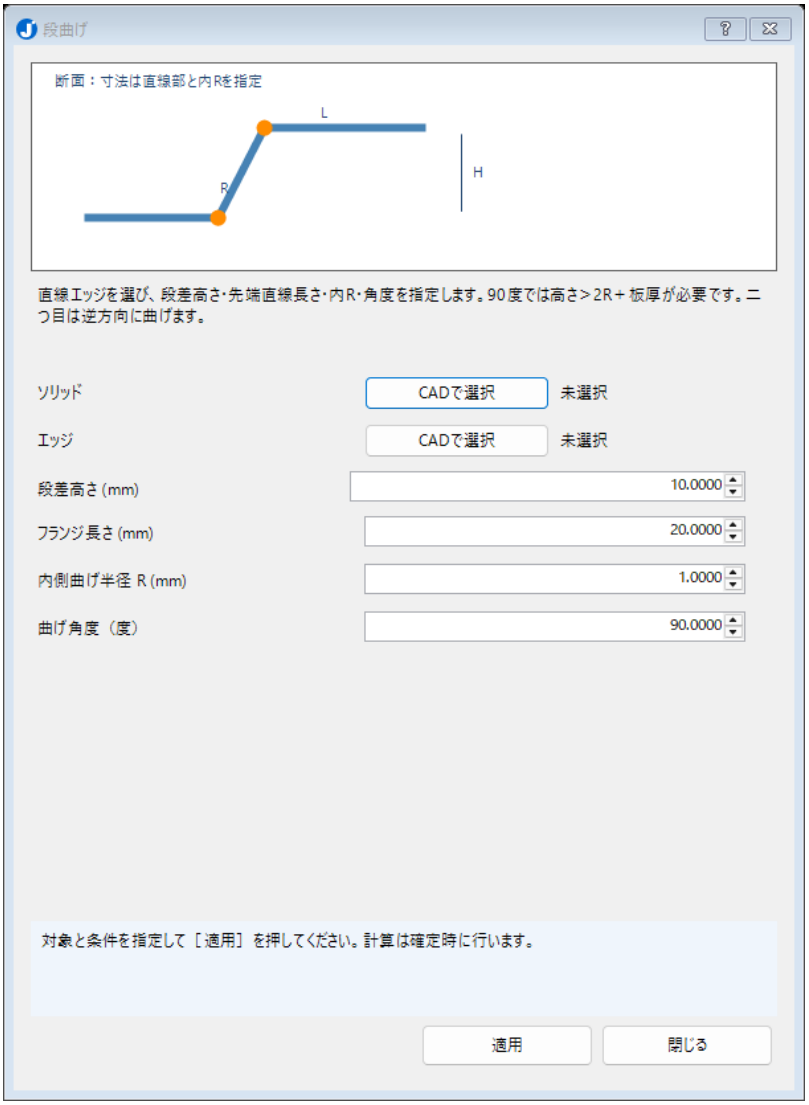


図 23　現在の操作画面。対象・入力値を設定して［適用］します。

板金として認識

一定板厚の平板を板金として登録します。ソリッド全体と、その広い基準面を選択します。
板厚は［適用］時に検出します。貫通穴は保持します。既存の曲げ付き形状は対象外です。

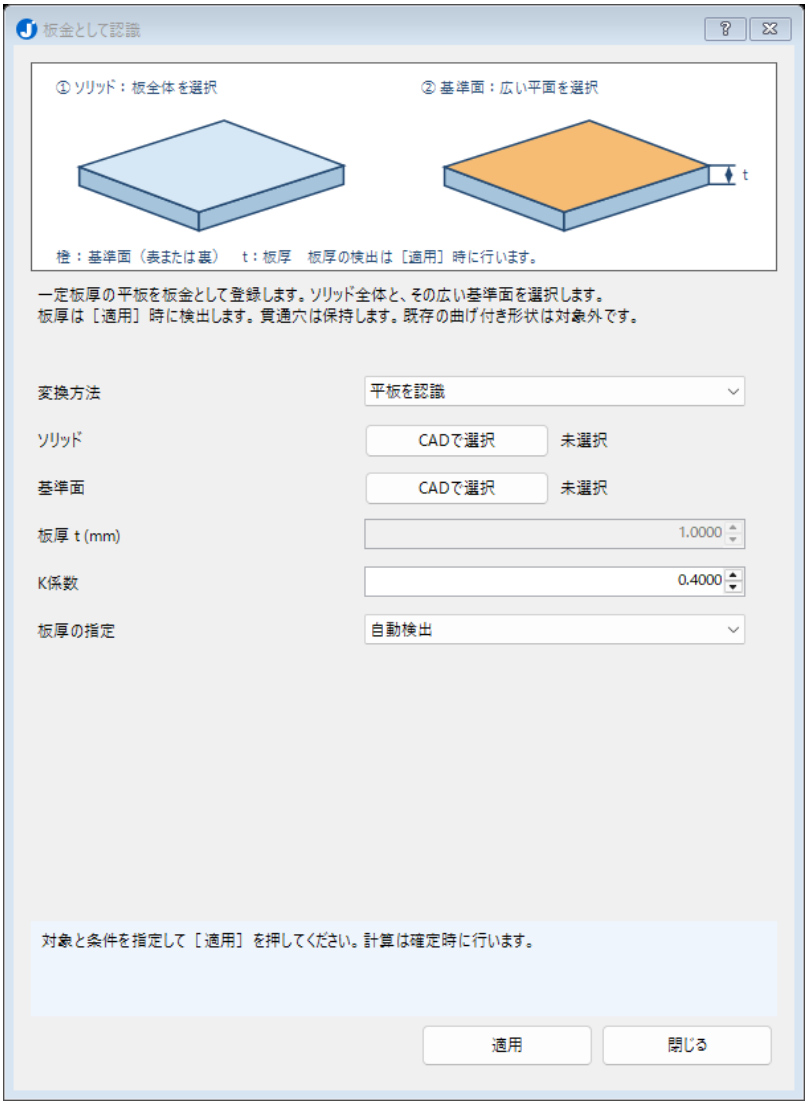


図 24　現在の操作画面。対象・入力値を設定して［適用］します。

再曲げ

展開中の輪郭編集と貫通加工を反映し、曲げ状態に戻します。曲げをまたぐ長穴にも対応します。
一定板厚で接続を保つ形状が対象です。止まり穴・板厚変更・分離・干渉する形状は適用しません。

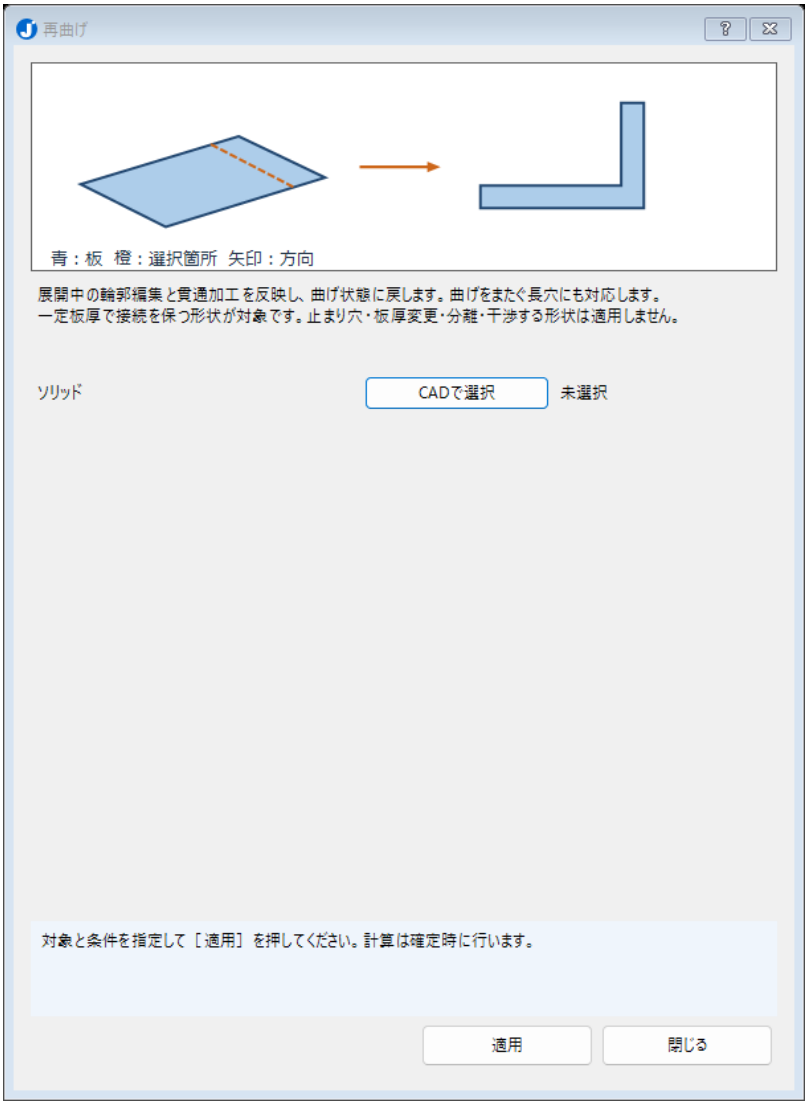


図 25　現在の操作画面。対象・入力値を設定して［適用］します。

切れ目を作成

箱の角を開く切れ目や、面上の線に沿う貫通スリットを作ります。種類を選び、ソリッドと対象を指定して［適用］します。

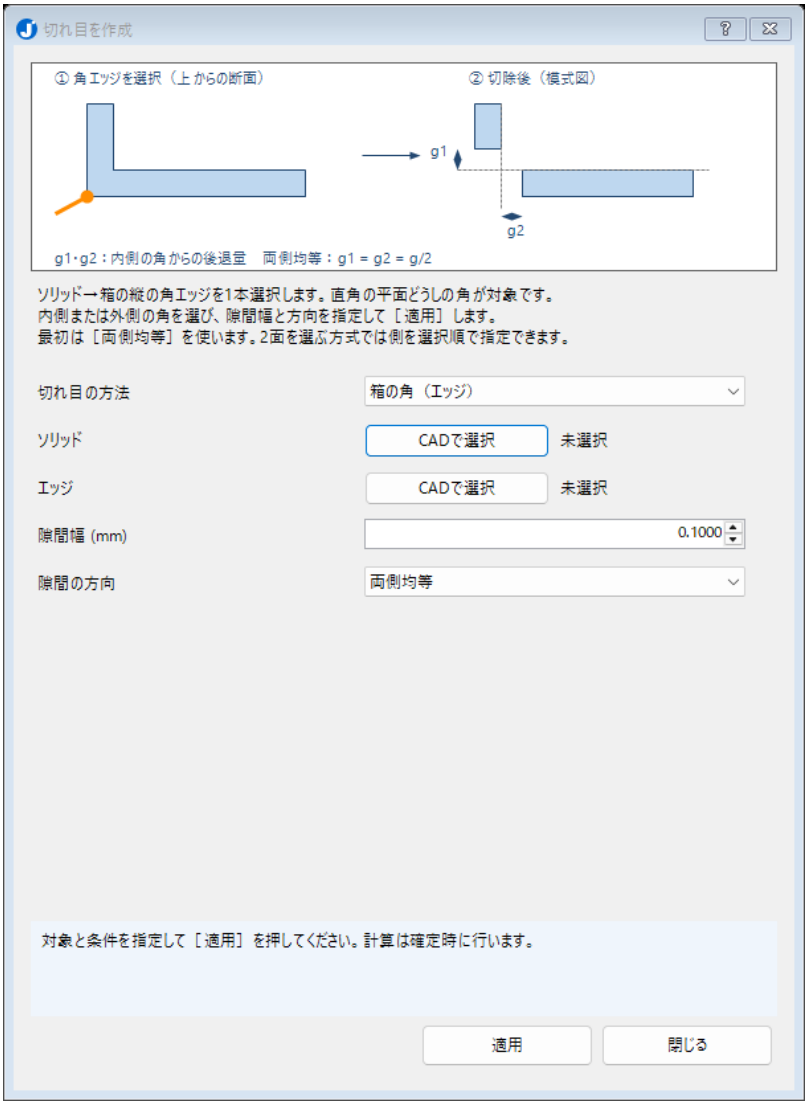


図 26　現在の操作画面。対象・入力値を設定して［適用］します。

ロール・ロフト

断面の種類を選び、寸法・高さを指定します。丸丸・角丸・角角の三角法展開に対応します。詳しくは［?］。



図 27　現在の操作画面。対象・入力値を設定して［適用］します。

板金設定

新規作成の既定値では、ソリッドの選択は不要です。

部品別に編集する場合は［設定対象］を［選択した板金部品］に変更して選択します。

保存だけでは形状は変わりません。新規フランジの入隅には逃げを自動作成します。

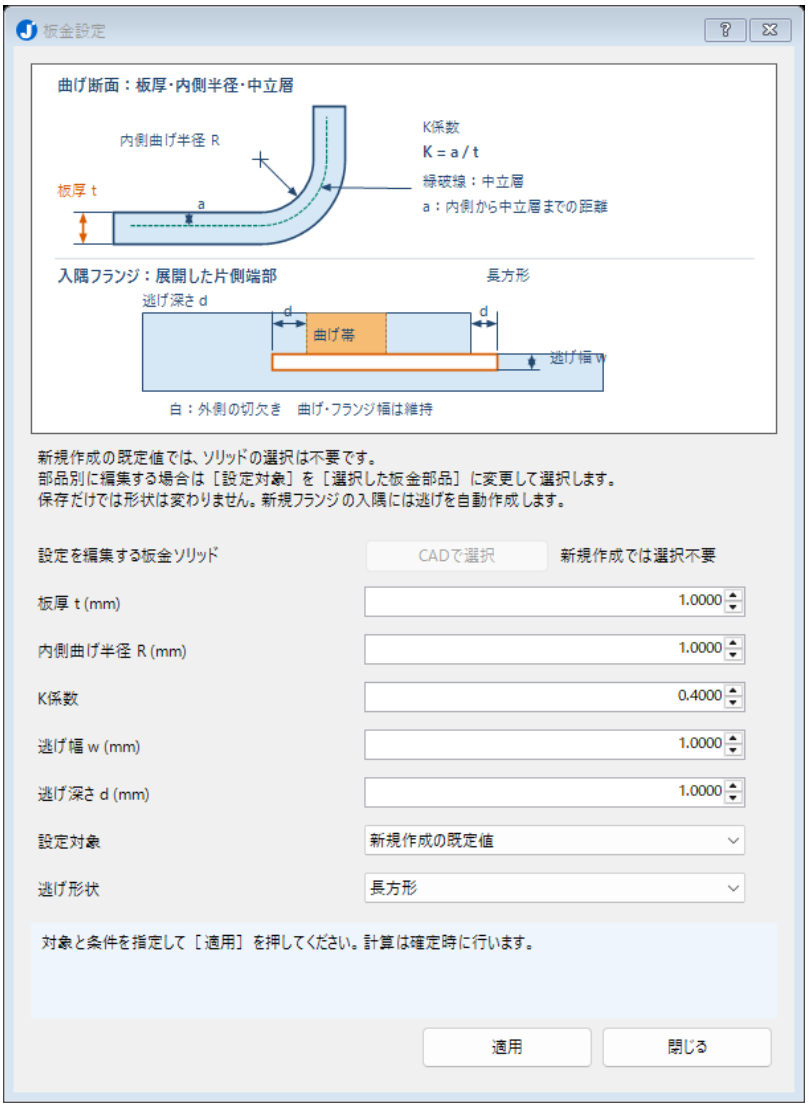


図 28　現在の操作画面。対象・入力値を設定して［適用］します。

展開

基準板の位置を保って全展開します。水色の曲げ帯中央線は、下書きの基準や端点・中点・線上のスナップに使えます。

通常コマンドで輪郭や曲げをまたぐ貫通穴・長穴を加工し、再曲げできます。

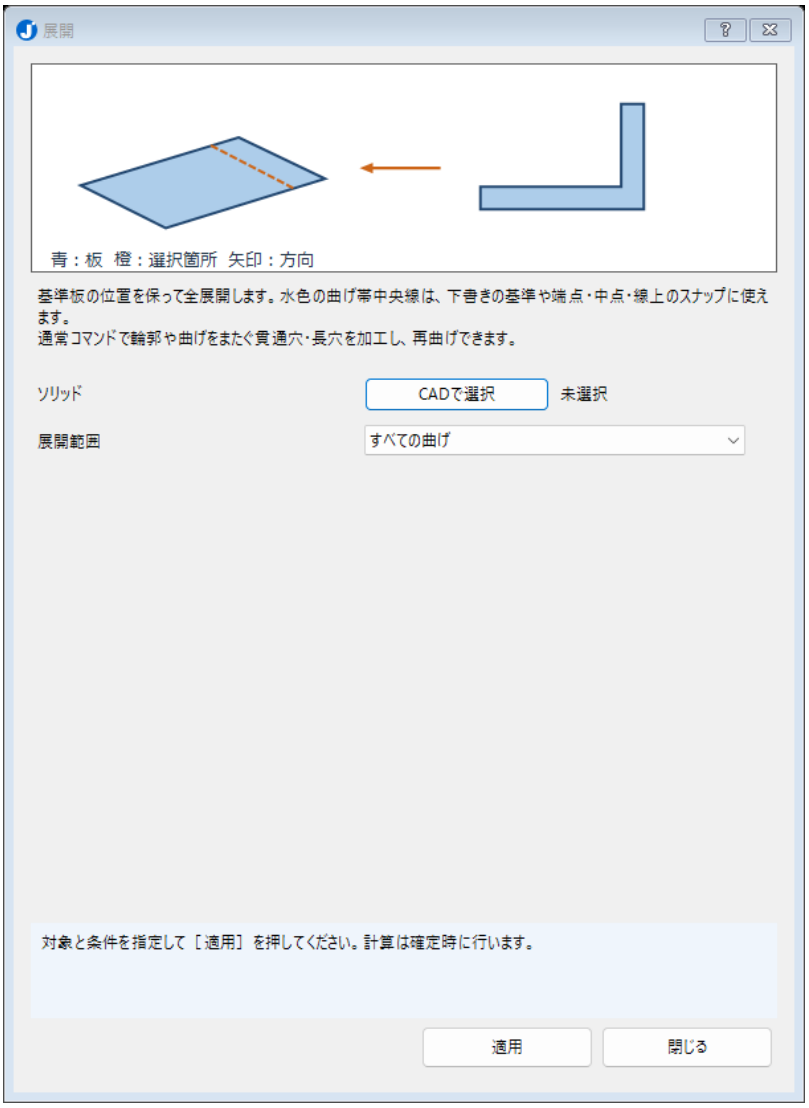


図 29　現在の操作画面。対象・入力値を設定して［適用］します。

板金設定の寸法を図で理解する

次の操作：板金として認識（対象・基準面・板厚の図解）輪郭から新規作成：基本板（板厚と厚みの方向）

切れ目作成：箱の角と面上のスリット

角箱を板金へ変換：底面・曲げ R・角の逃げ幅

通常コマンドで編集：幅調整・長穴・台形フランジ

エッジフランジ：既存フランジへの追加・範囲指定

曲げ逃げ：新規作成・既存の変更・削除

1. 最初に行う「板金設定」

ここで保存するのは、新しく作る板金の条件です。設定だけでは板や逃げ形状は作成されません。

• 先に「設定対象」を「新規作成の既定値」にします。この場合、ソリッドは未選択で構いません。

• 板厚 t、内側曲げ半径 R、K 係数、逃げ幅 w、逃げ深さ d、逃げ形状を入力します。

• 「適用」→「閉じる」の順に押し、図面を保存します。

• 閉じた輪郭から「基本板」を作り、「エッジフランジ」などで曲げます。既存の平板ソリッドなら「板金として認識」を使います。

• 新規のエッジフランジでは、周囲の板につながる入隅の端部に、部品の逃げ設定を使って自動作成します。自由端には追加しません。「逃げなし」では自動作成しません。

作成済みのフランジへ追加・変更する場合や、自由端へ逃げを付ける場合は「曲げ逃げ」で端部の円弧エッジを選びます。既存図面を開くだけでは逃げは追加されません。既存部品の次のフランジに使う値は、「新規作成の既定値」ではなく「選択した板金部品」で更新します。

「設定を編集する板金ソリッド」は、既存の板金部品の条件を編集するための選択欄です。新規作成の既定値では「新規作成では選択不要」と表示され、選択ボタンは無効になります。

既存部品の条件を更新する場合は「設定対象」を「選択した板金部品」に変更します。「編集する板金を選択」の案内に従い「CAD で選択」→数値入力→「適用」の順に進みます。平板でも、この板金機能で曲げた部品でも選択できます。普通の平板ソリッドは、先に「板金として認識」で板金部品にしてください。

対象を切り替えたり選択したりすると保存済み条件を読み込むため、数値はその後に入力してください。選択だけでは形状は変わりません。適用しても既存部品の板厚は変更できず、作成済みの曲げ形状も再計算されません。

2. 板厚 t と内側曲げ半径 R

R は板の内側の円弧半径です。外側の半径は R＋t になります。同じ板厚でも、R/t が小さいほど内側が小さく曲がり、大きいほどゆったりした丸みになります。

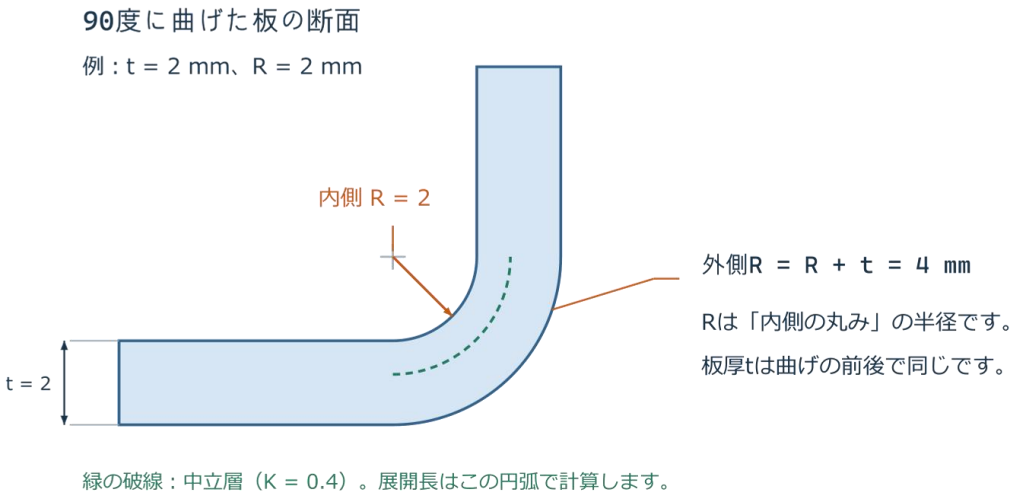


図 30　板厚 2 mm、内側 R2 mm の断面。外側 R は 4 mm。緑破線は中立層。

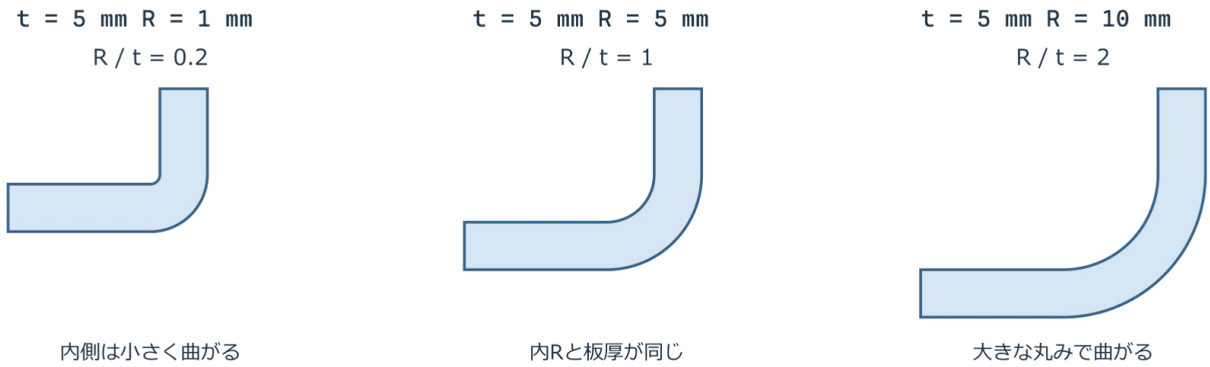


図 31　同じ板厚 5 mm で、内側 R1、5、10 mm を同縮尺で比較。R/t は 0.2、1、2。

これは形の違いの比較です。製造できる最小 R は、材質・調質・圧延方向・金型・曲げ方法などで変わります。R=t をすべての材料に適用できる製造条件と扱わないでください。

補足. 入隅のフランジは外側に逃げを作る

「逃げなし」と同じ幅の曲げ・フランジを作り、その外側の隣接する板を切り欠きます。曲げ R だけを細くしたり、立ち上がりの根元に段差を付けたりしません。長方形も丸形も同じ配置です。

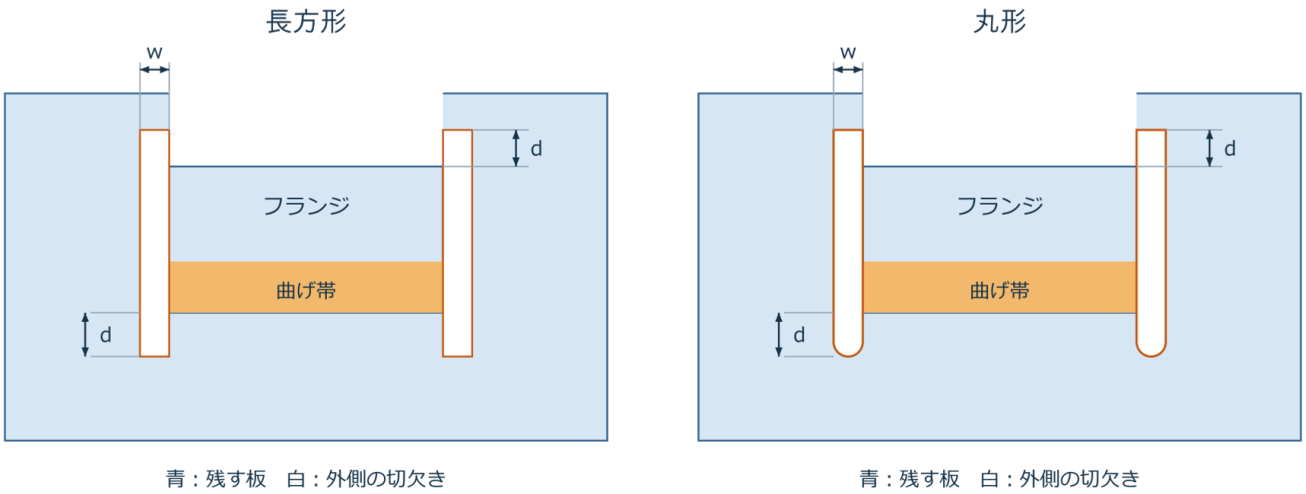


図 32　展開状態のコの字形の板。フランジと曲げ帯の幅は同じで、両側の板に幅 w の切欠き。丸形は基準板内の奥が半円。

w はフランジの外側へ切る幅、d は曲げの始まりから基準板へ入る深さです。展開して隣の板とつながらないよう、切欠きの開口側は展開したフランジ先端より d だけ先まで抜きます。曲げ状態でより長い隙間が必要な寸法では、その接触範囲も含めます。板がない部分は削りません。

丸形の奥は半径 w/2 の半円で、d は半円の最遠端まで。開口側を外周に接する半円にはしないため、尖った端部を残しません。形状が分離する寸法は適用できません。

新しいフランジは部品の設定で自動作成します。旧版で保存した入隅の逃げは、開く・展開するだけでは変更しません。「曲げ逃げ」で両端の円弧を選び、同じ寸法で適用するとこの方式に更新します。

3. 逃げ幅 w・逃げ深さ d を測る位置

以下は自由端・線曲げに使う従来方式の図です。入隅フランジの外側の逃げは、上の補足図を参照してください。

w は板の側端から内側へ切り取る幅、d は曲げ帯の境界から平面部へ延ばす長さです。下図の薄い橙色の部分を板厚いっぱいに切り取ります。d は板厚方向の穴深さではありません。

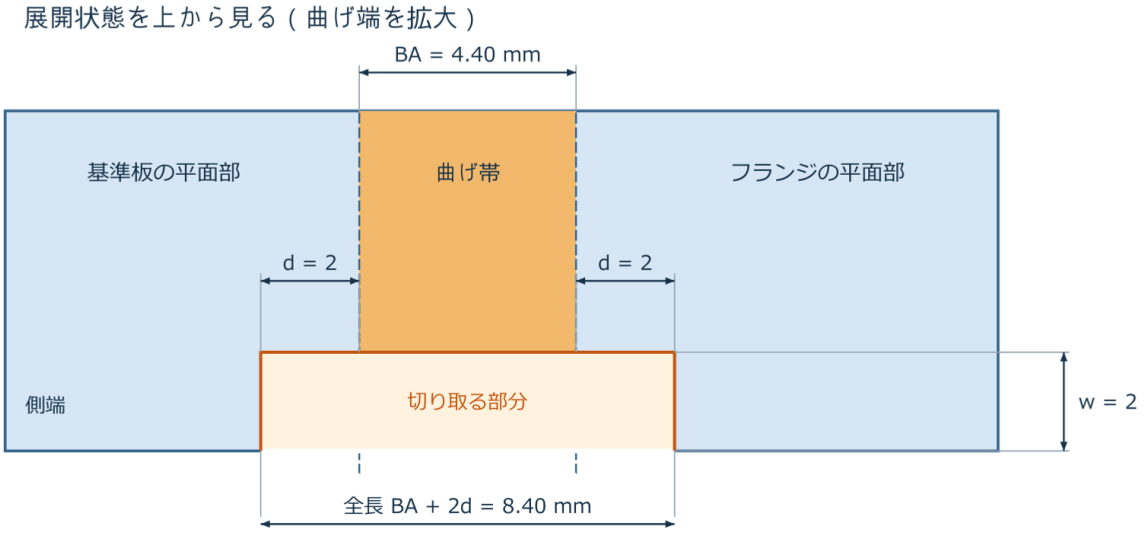


図 33　展開した板の曲げ端。幅 w は側端から内側へ。深さ d は曲げ帯の両境界から左右の平面へ。

JDesignCAD の現在の実装では、同じ d を基準板側とフランジ側の両方に適用します。切欠きの全長は BA＋2d です。BA は曲げ部分を展開した長さで、90 度なら BA＝π/2 × (R＋K×t)です。

図の例：t=2、R=2、K=0.4、w=2、d=2 mm → BA=4.40 mm、切欠き全長=8.40 mm。幅 w を増やすと曲げ端がより内側へ退き、d を増やすと切欠きが両方の平面部へ長く延びます。

この d の基準は JDesignCAD の定義です。他の CAD で同名の「逃げ深さ」があっても、寸法の起点や片側・両側の扱いを確認してください。

4. 長方形と丸形の違い

以下は自由端・線曲げに使う従来方式の図です。入隅フランジの外側の逃げは、上の補足図を参照してください。

丸形は切欠きの奥の角を R でつなぎ、入口を板の外周へまっすぐ抜く形です。外周に円を接する旧形状は、薄い尖った先端が残るため修正しました。下図は修正後の形です。

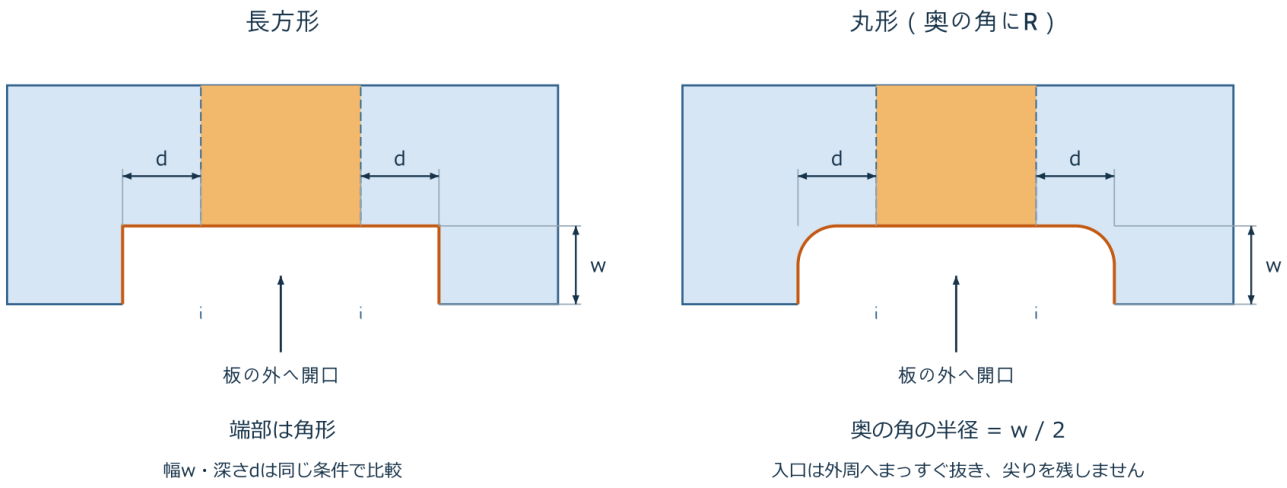


図 34　長方形と修正後の丸形。丸形は奥の角を半径 w/2 の四分円でつなぎ、入口を外周まで直線で抜く。

長方形は奥の角が直角、丸形は奥の角に半径 w/2 の四分円を付けます。この半径は、入力する曲げ半径 R とは別の寸法です。深さ d は曲げ帯の境界から切欠きの最遠端までの全体寸法で、丸みの半径を追加で足す必要はありません。

白い部分は板の外へ開いた切欠きです。板の内部に閉じた長穴を開ける図ではありません。「切欠きの奥のコーナーを丸める」というイメージです。角の R は w/2 で決まり、現状は独立して指定する項目はありません。

旧版で作成済みの丸形は、ファイルを開くだけでは置き換わりません。修正版で展開・再曲げするか、曲げ逃げを同じ条件で適用すると、新しい輪郭で再生成します。

また、板の外周コーナーを丸める処理や、2 つの曲げが交わる角に円形の切欠きを設ける「コーナー逃げ」とも区別してください。この図は 1 つの曲げの端に設ける「曲げ逃げ」を説明しています。

現在の入力条件：w は曲げ軸の長さの半分未満、d はフランジの直線長さ未満。丸形ではさらに d > w/2 が必要です。例えば w=2 mm なら、d=1 mm ちょうどは不可、d=2 mm はこの条件を満たします。d が w/2 に近い場合など、数値条件を満たしても形状検査で適用できない寸法があります。これらは形状処理の条件で、製造可否を保証する条件ではありません。

5. ダイアログの図と「逃げなし」

「板金設定」と「曲げ逃げ」の上部の図は、入力欄の板厚 t・内側曲げ半径 R・K 係数・逃げ幅 w・逃げ深さ d がどの寸法かを示します。具体的な数値を入れない模式図で、入力値に合わせて形状が伸縮するプレビューではありません。数値欄にカーソルを移すと、図の対応する寸法を橙色で強調します。緑破線は中立層で、a は内側から中立層までの距離、K=a/t です。



図 35　逃げなし：端部に切欠きがなく、幅 w と深さ d を使用しない。

逃げを不要とする場合は「逃げ形状」で「逃げなし」を選びます。w・d の入力欄は無効になります。幅や深さをゼロにする必要はありません。保存済みの数値は保持され、「長方形」か「丸形」に戻すと再び入力できます。

• 「板金設定」→設定対象を指定→「逃げなし」→「適用」。設定だけでは既存形状は変わらず、その条件で次に作るエッジフランジには逃げを自動作成しません。

• 既存の逃げを消す場合は「曲げ逃げ」→対象部品と端部の円弧エッジを選択→「逃げなし」→「適用」。選択した端部の逃げだけを取り除きます。両端とも消すなら両端を選びます。

• この板金機能で作成した逃げが対象です。任意の穴や切欠き、コーナー処理の隙間を埋めるコマンドではありません。

「逃げなし」は切欠きを作らない形状指定です。材料が実加工で裂ける様子や塑性変形を再現する「ティア」のシミュレーションは行いません。

6. 板厚に対する数値の読み方

以下は自由端・線曲げに使う従来方式の図です。入隅フランジの外側の逃げは、上の補足図を参照してください。

形を把握するため、R=w=d=t、K=0.4、曲げ角度 90 度にそろえた例です。製造推奨値や最小値の一覧ではありません。

板厚 t	内 R	K	逃げ幅 w	逃げ深さ d	90 度の BA
1 mm	1 mm	0.4	1 mm	1 mm	2.20 mm
2 mm	2 mm	0.4	2 mm	2 mm	4.40 mm
3 mm	3 mm	0.4	3 mm	3 mm	6.60 mm
5 mm	5 mm	0.4	5 mm	5 mm	11.00 mm

先ほどの画面の t=5、R=1、w=1、d=1 mm は、R/t=w/t=d/t=0.2 です。「5 mm 厚の板に、1 mm の内 R と 1 mm 幅の小さい逃げを指定している」と読めます。外側 R は 6 mm、90 度の BA は 4.71 mm、切欠き全長は 6.71 mm です。

K は板厚だけから決まりません。形状確認時は同じ値で比較し、加工用の展開寸法は材料・板厚・金型・曲げ方法に対応した実測値や加工業者の曲げ表に合わせます。

7. 材料の代表例と次の操作

候補例は SPCC（冷延鋼板）、SECC（電気亜鉛めっき鋼板）、SUS304（ステンレス）、A5052（アルミ合金）です。材料名だけで R や K を確定せず、同じ材料でも板厚や調質などを区別して条件を決めます。現在の設定画面に材料を選ぶだけで条件を自動決定する機能はありません。

板金設定 → 基本板／板金として認識 → 曲げ作成 → 必要に応じて曲げ逃げ → 展開 → 平面部に貫通穴などの加工 → 再曲げ

展開・再曲げの対象はこの板金機能で作成した曲げです。現在は曲げ帯を横切る切削、止まり穴、材料追加、板厚変更には対応していません。

根拠と資料

JDesignCAD 固有の w・d の基準は SheetMetalCuts.cs、BA は SheetMetalBend.cs、保存範囲は SheetMetalCommandAction.cs の実装を確認。一般的な用語と加工条件の考え方は次の資料を参照。図は本ヘルプ用に作図。

- Autodesk Fusion: Sheet metal rule reference
- Autodesk Inventor: About Bend Tables for Sheet Metal Materials
- Protolabs: Sheet Metal Bend Radius Guidelines

板金として認識：選び方と操作の順番

読み込み形状の認識と通常編集

［板金として認識］→［既存の円筒曲げを認識］で、広い固定平面を選択します。接線でつながる平面と円筒をたどり、内外の板厚を確認します。穴・切欠きと段ごとの角度を保持し、再構築した全外形が一致する場合だけ認識します。閉じた輪をなす形状は先に継ぎ目を開きます。自由曲面のプレス成形は対象外です。

箱変換は高さが異なる直角の側板にも対応します。通常の［面の移動］では底面の側辺に沿う曲げ軸を持つ箱の幅を変更でき、側板の角度は 90 度以外でも維持します。移動した側板の成形も追従します。底面・横側板の輪郭は移動方向に一つの連続領域であることが必要です。穴の大きさは変えません。

平面部の面移動・テーパーと部分展開中の平面編集は、板厚と接続を維持する範囲で使用できます。通常のミラーも板金情報と表裏を保持します。拡大縮小による板金寸法の変更は使用せず、板厚・曲げ・面編集を使ってください。

板金設定のヘルプへ · English · 印刷用 PDF（日本語・4 ページ）

1. このコマンドで行うこと

［変換方法：平板を認識］では、すでにある、一定の厚さの平板ソリッドを「板金部品」として登録します。モデル空間の同じ位置で、外形・板厚・貫通穴を保ったまま、以後の板金操作に使う情報を付けます。適用しただけで曲がったり、板厚が変わったりはしません。

通常の押し出しなどで作った平板や、読み込んだ平板ソリッドから板金作業を始めるときに使います。「基本板」で作成したものには最初から板金情報があるため、認識は不要です。線だけの輪郭から板を作る場合は「基本板」を使います。

2. ソリッドと基準面の選び方

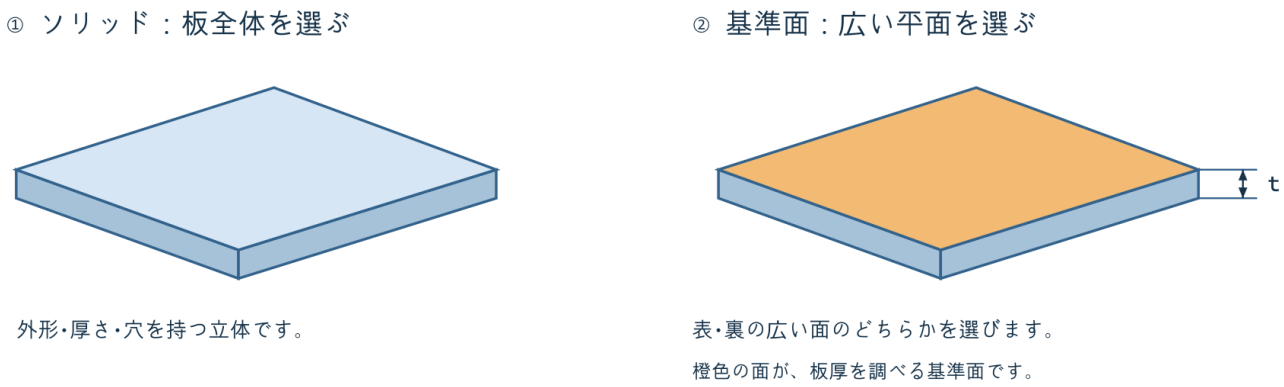


図 36　左は板全体のソリッド、右はオレンジ色の広い基準面。t は板厚。

- 「板金として認識」を開きます。
- 「ソリッド」の「CAD で選択」で、認識させる板全体を選びます。
- 「基準面」の「CAD で選択」で、同じ板の広い平らな面を選びます。表・裏のどちらでも構いません。

基準面は、実際の板厚を調べるための面です。細い側面、穴の円筒面、他の部品の面は選びません。箱状の形では、細い側面を選ぶと意図しない方向を板厚として解釈する場合があります。基準面の選択だけで、後の曲げ位置や曲げ角度を指定するわけではありません。

ソリッドを選び直すと、前の基準面の選択は解除されます。対象のソリッド→基準面の順に選んでください。

3. 板厚の指定・K 係数・適用

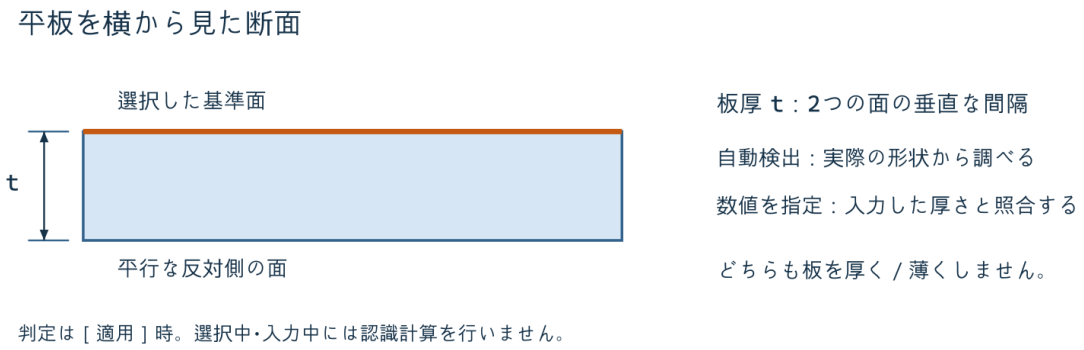


図 37　平行な表裏の面の垂直距離が板厚 t。自動検出は測定、数値を指定は一致の確認。

項目	意味と使い方
板厚の指定：自動検出	通常はこちらを使います。板厚欄は入力できず、［適用］時に実形状から厚さを調べます。選択中のグレー表示は、測定結果ではなく初期値・読み込み済み条件です。
板厚の指定：数値を指定	板厚欄へ意図する厚さを入力します。［適用］時に実際の厚さと一致するか確認します。不一致なら適用しません。板を厚く／薄くする操作ではありません。
K 係数	次に作る曲げの展開長を計算する条件です。K=a/t で、a は将来の曲げの内側から中立層までの距離です。認識時点で平板の形は変わりません。材料名だけから自動決定はしません。

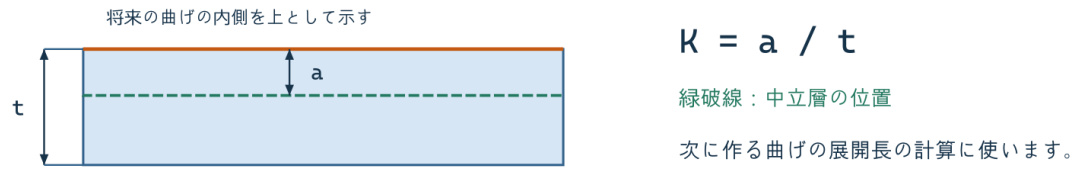


図 38　板厚 t の中で内側から中立層までを a とし、K=a/t と定義する。

- 通常は「板厚の指定」を「自動検出」にし、K 係数を確認します。
- ［適用］を押します。この時点で初めて認識計算と形状検査を行います。
- 成功したら［閉じる］を押し、図面を保存します。

板厚は実形状に基づく値を登録します。登録値を確認するには「板金設定」→「設定対象：選択した板金部品」で部品を選びます。長さの入力単位は欄に表示された単位に従います。K 係数に単位はありません。

4. 認識できる形・対象外の形

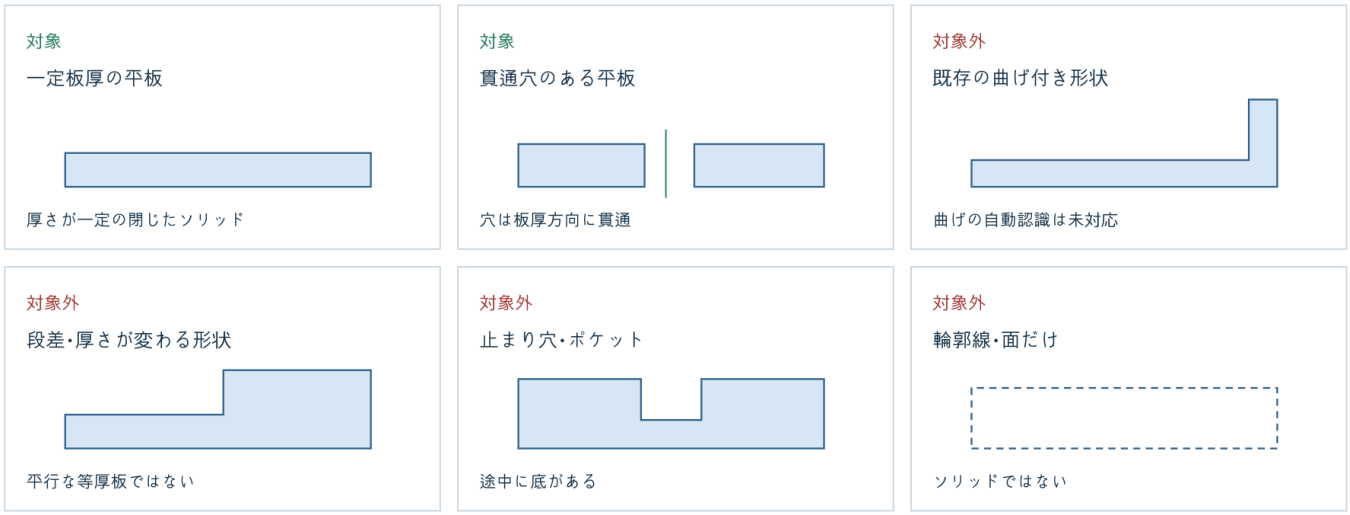


図 39　一定板厚の平板と貫通穴のある平板が対象。既存曲げ、段差、止まり穴、線や面だけの形状は対象外。

図は代表的な断面例です。閉じたソリッドであること、表裏の面と境界が対応すること、一定の厚さであることを検査します。貫通穴や板厚を貫く切欠きがあっても、平板として条件を満たせば対象です。薄い板でも、段差・テーパー・板厚方向の面取りや丸みがある形状は、この初期版では認識対象から外れる場合があります。

他の CAD で曲げ済みの板金部品を、そのまま自動展開する機能ではありません。現在は平板から登録し、JDesignCAD の板金コマンドで曲げを作る手順です。

箱の角に切れ目を入れただけでは、曲げ板金へ変換されません。通常ソリッドの切れ目作成は形状を切削する操作です。箱を展開可能な板金へ変換するには、固定面・曲げ位置・板厚・内側曲げ半径・展開条件を定義し、必要な逃げと曲げ形状を作る別の処理が必要です。底面を固定して 4 枚の側板を開く構成では、側板どうしの縦 4 隅を切り離します。[変換方法：箱・L 形・コの字を板金へ変換] は、一定板厚の長方形の底と側板 1〜4 枚に対応します。角の縦切れ目がある形も対象です。穴・段差・既存の曲げ R は対象外です。角箱の変換手順と寸法の図解を参照してください。

5. 適用後の操作と、うまくいかないとき

平板ソリッド → 板金として認識 → エッジフランジ／線で曲げる → 展開 → 平面部の貫通加工 → 再曲げ

認識に成功しても見た目は同じです。同じモデル空間に板金情報が登録され、板金コマンドで操作できるようになります。部品の数は増えません。適用は「元に戻す」で取り消せます。検査に失敗した場合は元の形状を残します。

状態・メッセージ	確認すること
基準面を選べない	先に対象ソリッドを選び、その同じ部品の広い平面を選びます。
一定板厚の平板が対象と表示される	曲がり、段差、止まり穴、テーパーがないか、閉じたソリッドかを確認します。
指定した板厚と実際の板厚が一致しない	数値と表示単位、基準面を確認します。厚さが不明なら自動検出にします。

状態・メッセージ	確認すること
登録された厚さが想定と違う	基準面に細い側面を選んでいないか確認します。実形状や読み込み時の寸法も確認します。
適用したのに見た目が変わらない	正常です。板金情報の登録なので、曲げは次のコマンドで作ります。

展開中の板金に「板金として認識」を再実行しないでください。現在の実装では板金情報を新規登録し直すため、既存の曲げ情報を引き継ぎません。展開後の穴加工から元の形へ戻すときは、そのまま「再曲げ」を使います。

基本板：輪郭から板を作る・厚みの方向

板金設定 · 板金として認識 · English · 印刷用 PDF（日本語・2 ページ）

1. 基本板とは

閉じた平面の輪郭線に板厚を付け、新しい板金ソリッドを作るコマンドです。モデル空間で作業します。元の輪郭はその位置に残ります。作成した板には板金情報が付くため、「板金として認識」をもう一度行う必要はありません。

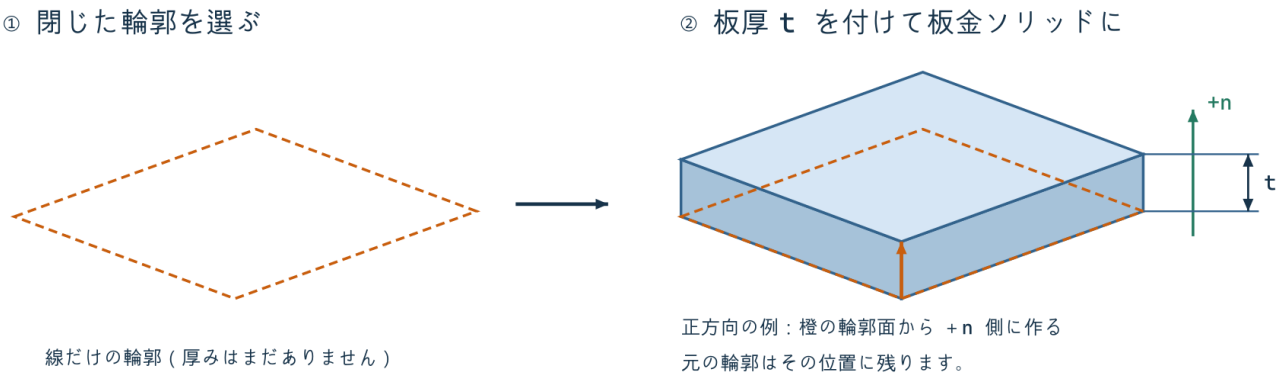


図 40　左は厚みのない閉じた輪郭、右は元の輪郭面からプラス n 方向へ板厚 t を付けた基本板。

2. 操作の順番

• 円・閉じたポリライン・閉じたパスなどで、同一平面上に 1 つの閉じた輪郭を用意します。別々の線の集まりは、先に 1 つの閉じたパスなどにまとめます。

• ［板金］→［基本板］を開きます。

• ［閉じた輪郭］の［CAD で選択］で、その輪郭を選びます。既存の板ソリッドやその面を選ぶ欄ではありません。

• ［板厚 t］へ、作成する板の全体の厚さを入力します。単位は入力欄の表示に従います。

• ［厚みの方向］を選びます。上の模式図がその方向に切り替わります。

• ［適用］で作成し、位置と厚さを確認して［閉じる］→図面を保存します。

選択や入力だけではソリッドを作りません。作成処理は［適用］時に行います。板厚に 0 や負の値は指定できません。

3. 正方向・負方向・両側均等

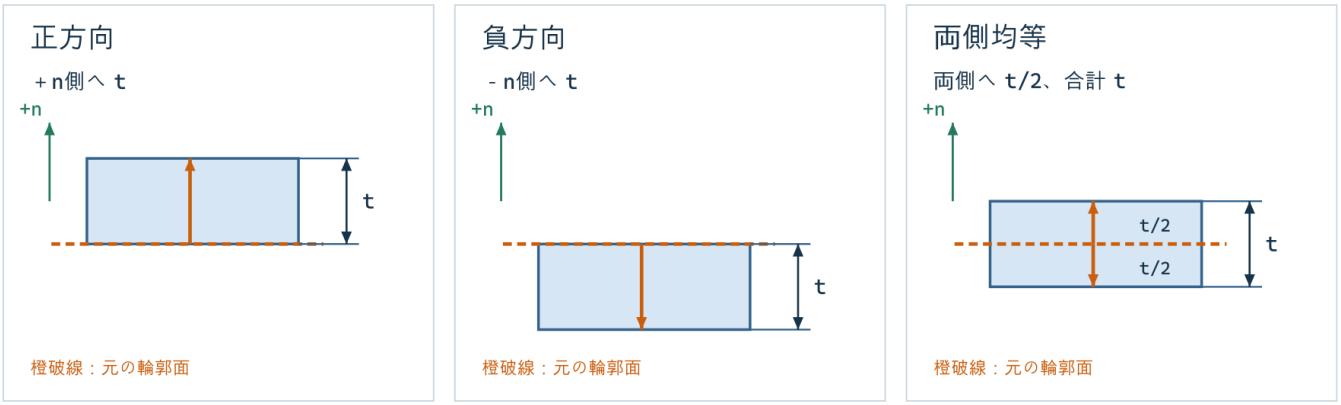


図 41　正方向は元の輪郭面からプラス n 側に t、負方向は反対側に t、両側均等は両側へ t の半分ずつ。

厚みの方向	輪郭面を基準に、どこに板ができるか
正方向	輪郭面から＋n 側に板厚 t を付けます。元の輪郭面が片側の表面になります。
負方向	輪郭面から－n 側に板厚 t を付けます。正方向とは反対側にできます。
両側均等	輪郭面を厚さの中心にして、両側へ t/2 ずつ付けます。全体の厚さは t で、2t にはなりません。

通常の XY 作業平面では「上（＋Z）が正方向、下（－Z）が負方向」です。輪郭を時計回り／反時計回りのどちらで描いても、押し出し方向は変わりません。内曲げ・外曲げの指定とは別です。

＋n は輪郭面に垂直な方向を、コマンド開始時の作業平面の＋Z 側にそろえたものです。作業平面を傾けた場合は、その座標の向きを基準にします。コマンド途中の画面回転で正負は入れ替わりません。作業平面に垂直な輪郭では＋Y 側、それとも垂直なら＋X 側を正方向とします。図は向きを示す模式図で、実際の輪郭形状や入力寸法のプレビューではありません。

意図と反対側にできた場合は、同じダイアログを開いたまま正方向／負方向を切り替え、再び［適用］します。直前にこのコマンドで作成した板が、新しい条件の板に置き換わります。

4. 作成後・うまくいかないとき

状況	対応
コの字型の板を作りたい	コの字型や L 字型など、凹みのある輪郭も使えます。外周が一周つながっていることが条件です。ポリラインは「閉じる」がオフでも、始点と終点が同じ位置なら選択できます。作成用のコピーを閉じるため、元の輪郭は変わりません。

状況	対応
輪郭を選べない	一周つながった 1 つの曲線オブジェクトか確認します。別々の線、端点に隙間のある線、ソリッドは対象ではありません。
適用できない	輪郭が同一平面上にあり、閉じていて、自己交差や不正な重なりがないか確認します。
穴のある板にしたい	現在は 1 つの輪郭から板を作ります。外周と穴の輪郭を複数同時に選ぶ方式ではありません。まず基本板を作り、その後に通常の穴加工などを行います。
作成後に厚さや方向を変えたい	同じダイアログを開いたままなら、値を変更して［適用］し、直前の基本板を置き換えられます。閉じてから新しく［基本板］を開始した場合は新規作成になります。以前の板が不要なら［元に戻す］などで整理してから作成します。

閉じた輪郭 → 基本板 → エッジフランジ／線で曲げる → 展開・加工 → 再曲げ

すでにある平板ソリッドから始める場合は「板金として認識」を使います。基本板を作っただけでは曲げ・逃げは作られません。

エッジフランジ：基準板・既存フランジから作成

複数辺のフランジ・折返し・段曲げ

［エッジフランジ］の［エッジ全体］では複数辺を選べます。範囲指定は 1 辺ずつです。一括作成中の一つでも失敗した場合は、全体を変更しません。

コマンド	指定方法
折返し（開きヘム）	180 度の折返し。隙間 g から内 R＝g/2 を決めます。L は折返し後の直線部です。複数辺に対応します。潰しヘム・巻込みヘムは含みません。
段曲げ	段差高さ H、先端直線部 L、内 R、曲げ角度を指定します。二つ目の曲げは逆方向です。90 度では H＞2R＋板厚が必要です。
連続フランジ	行ごとに相対角度・直線部 L・内 R を指定します。負の角度は逆方向です。最下行に追加し、行頭を選んで Delete で削除します。

180 度では仮想交点の寸法が無限になるため、直線部の長さを使用します。

板金設定 ・English

箱の上部 4 辺を 45 度で合わせる手順

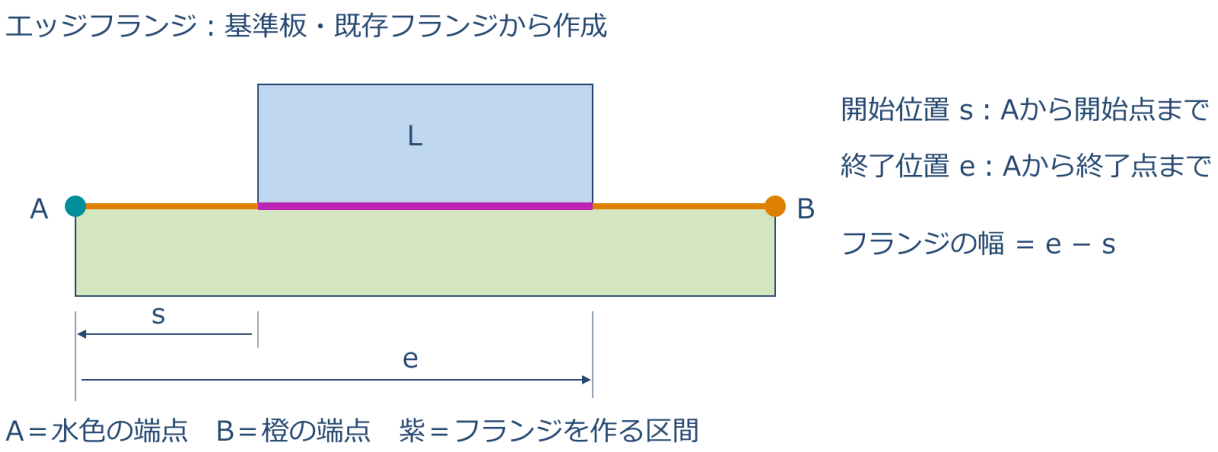


図 42　A＝水色の端点　B＝橙の端点　紫＝フランジを作る区間

加工後の再編集・自動 45 度合わせ・寸法基準の使い方

基準板だけでなく、既存フランジの先端・側辺にも新しいフランジを作れます。さらにそのフランジから続けて曲げを追加できます。エッジ全体と、途中だけの範囲指定に対応します。モデル空間の板金ソリッドで操作します。

操作の順番

- ［エッジフランジ］を開き、対象のソリッドを選択します。

• 基準板または既存フランジの、自由な外周の直線エッジを 1 本選びます。ソリッドは緑、選択エッジはオレンジになります。

• 全幅なら［エッジ全体］を使います。一部だけなら［フランジの範囲］で［開始・終了位置を指定］を選びます。

• ［開始位置（A から)］と［終了位置（A から)］へ距離を入力します。またはそれぞれの［点を指示］を押し、CAD 画面で位置を指示します。

• 長さ L、曲げ角度、内側曲げ半径を指定し、［適用］を押します。

既存フランジの先端・側辺へ追加

例えば、立ち上がった側板の上端を選ぶと、その先に折り返しを作れます。線で曲げた側板も対象です。ソリッド全体を選んだ後、追加したい側板の直線エッジを選んでください。すでに曲げにつながっている境界線は自由端ではないため選べません。

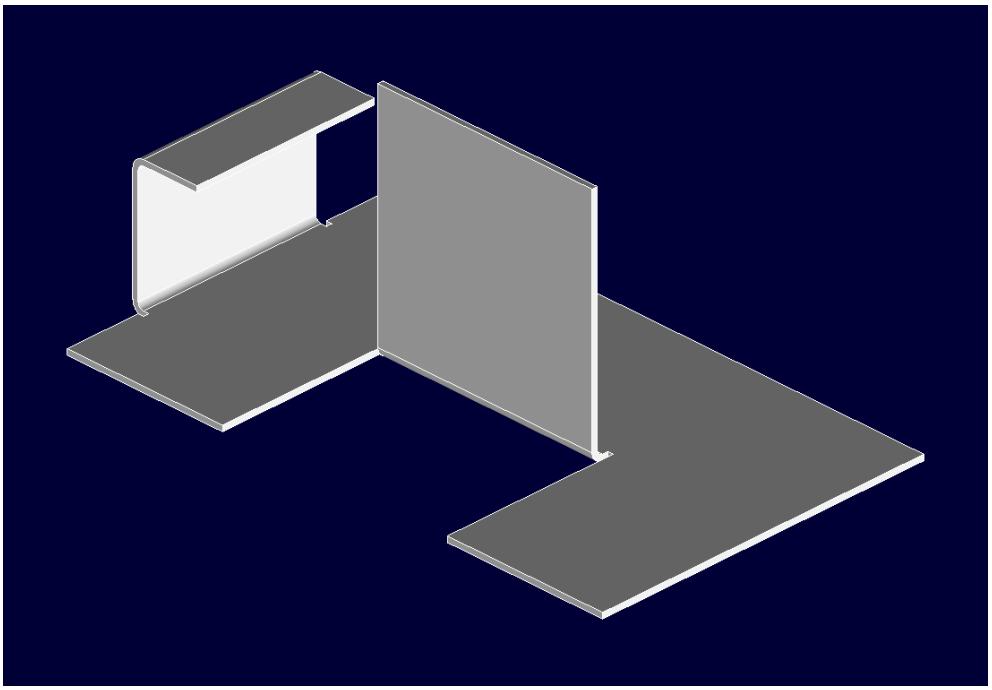


図 43　既存の立ち上がりフランジの先端へ折り返しを追加した実際の作成例

作成例：左の側板の上端から、さらにフランジを追加しています。

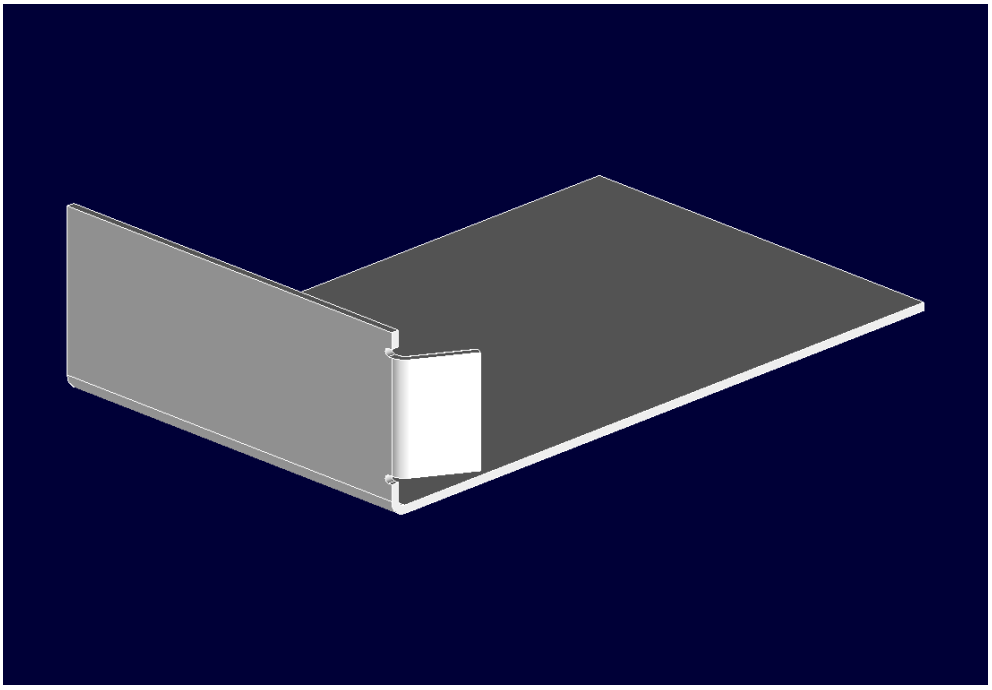


図 44　既存フランジの側辺の一部にフランジを追加した作成例

作成例：側辺の途中を指定した追加。端の逃げは板金設定に従います。

追加したフランジは接続元の側板に追従します。元の曲げの角度・半径を編集した場合も接続を保ちます。通常の長方形フランジの長さを変えると、先端に付けたフランジは先端とともに動きます。側辺の途中への取り付け位置は、曲げの接線からの距離を保ちます。短くして取り付け範囲がなくなる場合や、他の板と干渉する場合は適用しません。

範囲と色の読み方

水色の端点が A、オレンジの端点が B です。開始・終了位置は、どちらも A から B へ向かって測った距離です。終了位置は「反対側の端からの距離」ではありません。開始位置を 0 以上、終了位置をエッジ長以下にし、開始＜終了になるようにします。作成区間は紫で表示します。図は寸法の位置を示す模式図です。

［点を指示］では、指示点を選択エッジの直線上へ投影して距離に換算します。端点や中点などへスナップできます。別のエッジへ選び直すと、範囲は新しいエッジの全長へ戻ります。

入力	意味
開始位置 s・終了位置 e	フランジを作る幅の範囲。幅は e − s です。
長さ L	曲げの接線から先の直線部分の長さ。エッジに沿った幅とは別です。
エッジ全体	端点 A から B までを使います。開始・終了位置の入力は不要です。

逃げ・展開・作成後

エッジ途中の両端には、板金ソリッドに保存された逃げ形状・幅・深さを使います。長方形と丸形に対応し、逃げなしなら切欠きを加えません。端点 A または B まで使う場合、自由端には余分な逃げを作りません。

選択や数値変更中にはソリッドを変更せず、［適用］時に作成します。作成後は［元に戻す／やり直し］、展開・再曲げ、保存・再読込に対応します。フォームの［？］からこの説明を開けます。

対象は基準板・既存フランジの自由な外周の直線エッジです。曲線エッジ、穴の内周、既存の曲げとの境界は対象外です。曲げ同士が干渉する範囲や、板を分断する逃げ寸法は作成できません。

箱の上部 4 辺を 45 度で合わせる

隣のフランジを長方形で追加すると、角で材料が重なり、作成できません。［端部の形状］で最初から台形にして作成します。平面部分だけを後からテーパーしても、全幅の曲げ R 同士が重なる場合があります。［開始・終了位置を指定］で R の端にも隙間を確保します。

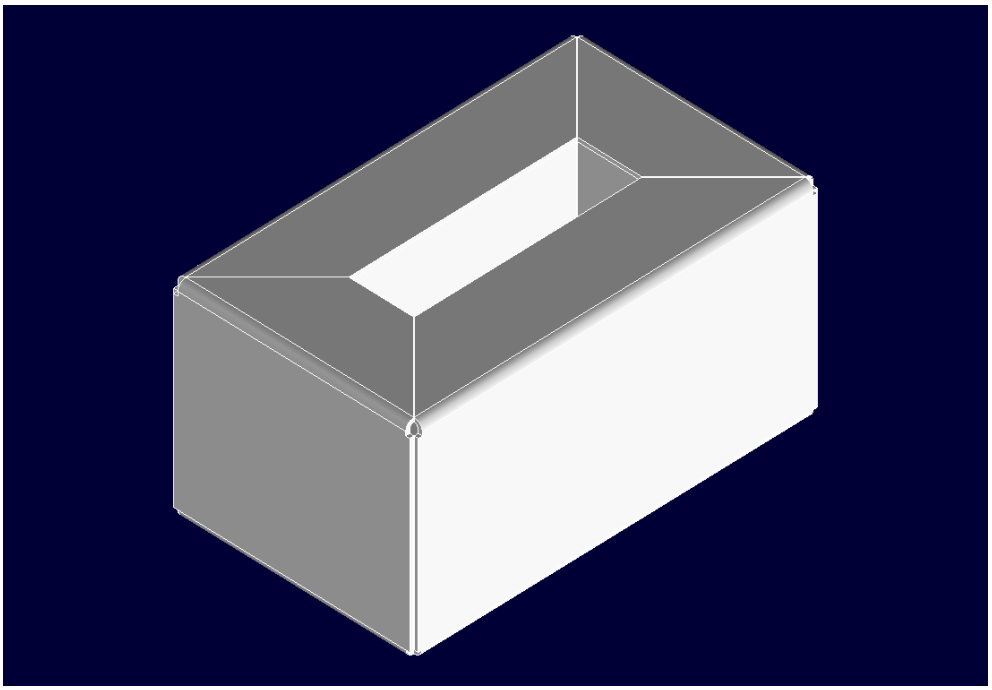


図 45　板金 18 の箱に、両端 45 度の内向きフランジを 4 辺作成した実際の結果

板金 18 から作成した例。4 辺とも平面部分を 45 度に狭め、曲げ R は両端を控えています。

板金 18 の入力例（板厚 1mm・R1mm・長さ 20mm）

上部フランジを付ける前の箱から始めます。すでに全幅の上部フランジを付けている場合は、元に戻して作り直すか、添付の作成前モデルを使ってください。［両端 45 度］を選ぶだけでは、既存フランジの根元は自動で短くなりません。

• ［エッジフランジ］で箱のソリッドを選び、側板上端の内側の直線エッジを選びます。内外で折れ曲がる方向が変わります。

• ［フランジの範囲］を開始・終了位置を指定、［端部の形状］を両端 45 度・先端を狭めるにします。

• 開始位置は 1mm、終了位置は選択エッジの長さ − 1mm にします。どちらも水色の端点 A からの距離です。

• 内側曲げ半径 1mm、フランジ長さ 20mm、曲げ角度 90 度で［適用］します。

• 残り 3 辺も、同じ設定でエッジを選び直して適用します。選び直すと開始・終了位置が全長へ戻るので、毎回入力し直してください。

板金 18 の選択エッジ	開始（A から）	終了（A から）
短辺：57.9mm	1.0mm	56.9mm
長辺：97.9mm	1.0mm	96.9mm



図 46　開始 1、終了 56.9、両端 45 度で先端を狭めるエッジフランジの設定

短辺の入力例。終了位置に 1mm と入力すると、幅 1mm の範囲になってしまうので注意してください。

この 1mm は板金 18 の寸法で確認した値です。板厚・R・壁の位置・角の隙間が違う場合は、同じ値で必ず合うわけではありません。実際の形状に合わせて範囲を調整してください。上の例の斜めの継ぎ目には約 0.07mm の隙間が残ります。曲げ R の端は直角に控えた形です。R 同士を斜めの曲面でつなぐ処理や、隙間の自動計算ではありません。

端部の形状と作成後の編集

選択肢	作成する形
直角（長方形）	従来と同じ一定幅です。
両端 45 度・先端を狭める	両端を内側へ 45 度にします。長さ L だけ進むと、各端が L ずつ狭まります。
両端 45 度・先端を広げる	両端へ材料を足し、先端を広げます。外向きフランジなどに使用できます。根元の幅は変わらず、隣のフランジとの隙間は別途調整します。

短辺を狭める場合、作成区間の幅が 2L 以下だと先端が交差するので適用できません。必要なら L を短くします。端部の逃げは板金設定の「なし・長方形・丸形」に従います。今回の 4 辺は 3 種類とも確認しています。

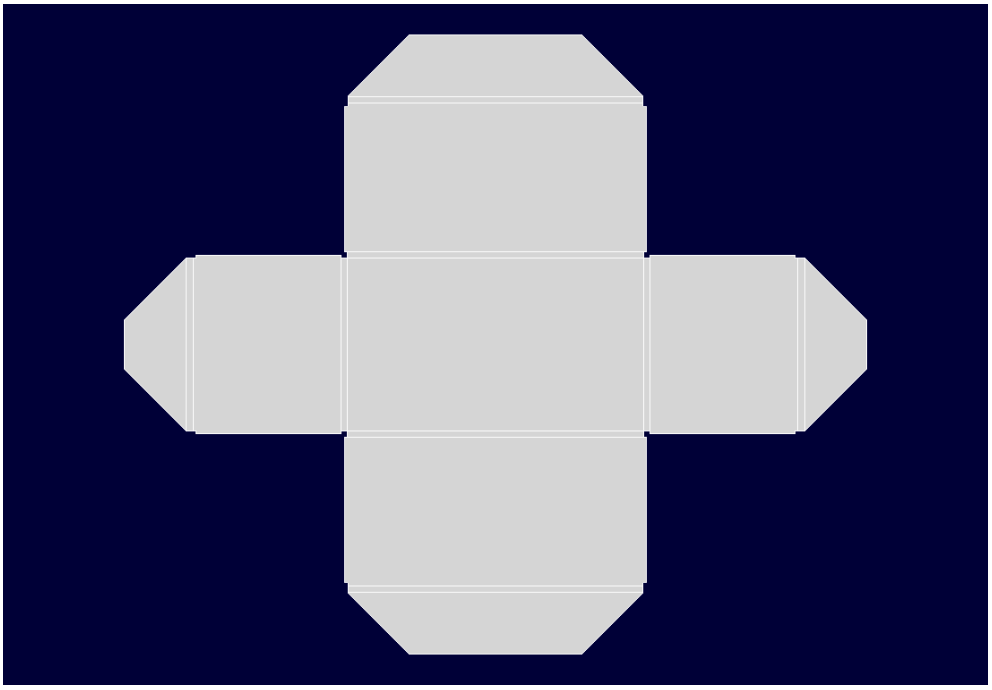


図 47　4 辺の台形フランジを展開した結果。曲げ帯の境界も残っている

展開結果。保存・再読込、再曲げ、元に戻す／やり直しを確認済みです。

作成後の端面の調整は通常の面のテーパーを使えます。ここで作る台形は、隣接するフランジを自動でトリムする機能ではありません。他の板へ干渉する形状は適用しません。

線で曲げる：既存の曲げがある板金

板金設定 ・English

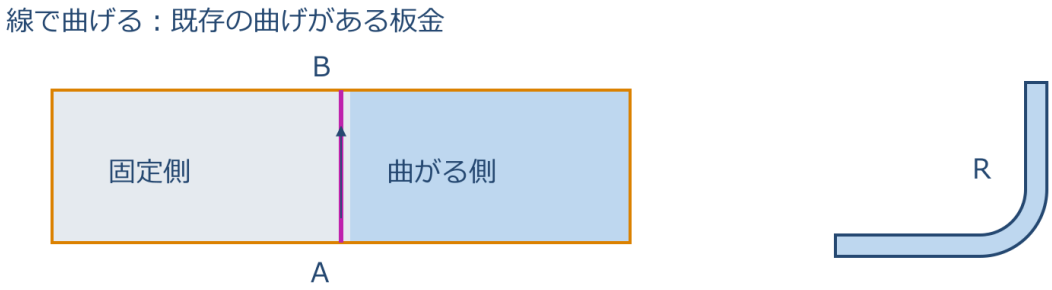


図 48　緑＝ソリッドの輪郭　橙＝固定面の輪郭　紫＝曲げ線

加工後の再編集・自動 45 度合わせ・寸法基準の使い方

平板だけでなく、エッジフランジや以前の「線で曲げる」がある板金ソリッドにも追加できます。曲げ線のある平らな部分を曲げ、移動側に付いている既存のフランジも一緒に動かします。

操作の順番

- 板金ソリッドの平らな面上に、端から端へ横断する線分を用意します。
- ［線で曲げる］を開き、ソリッドを選択します。緑の輪郭だけが強調されるので、面上の線分が見えます。
- ［固定面］で曲げ線のある平らな面を選びます。輪郭がオレンジになります。
- ［曲げ線］で線分を選びます。選んだ線は紫になります。
- 内側曲げ半径 R と曲げ角度を設定し、［適用］を押します。

どちらが動くか

選択した面を正面から見て、線分の始点 A→終点 B の右側を曲げます。左側が固定側です。固定面を選んだ場所ではなく、線分の向きで左右を決めます。逆側を曲げる場合は元の線分の向きを逆にします。

曲げ線から動く側へ、板厚・内側半径・K 係数・角度で計算した曲げ帯を確保します。既存のフランジが動く側に付いていれば、その取り付け位置と形状を保ちながら一緒に動かします。

確認できることと制限

- 基準板の平らな部分、既存フランジの平らな部分、以前の線曲げで動いた平らな部分に追加できます。
- 展開・再曲げ、保存・再読込、元に戻す／やり直しに対応します。展開後に平らな部分へ開けた貫通穴は、対応する面と一緒に再曲げされます。
- 既存の R 面そのものをもう一度曲げたり、曲げ帯が穴や既存の曲げの付け根を横切る指定には対応しません。
- 曲げ線が板を横断しない場合や、曲げた後にフランジ同士が干渉する場合は作成しません。展開中の部品は先に再曲げしてください。

選択・入力中は形状を計算せず、［適用］時に作成します。計算に失敗した場合は元のソリッドを保持します。

曲げ・フランジ編集

曲げ一覧・抑制・削除・親変更

［曲げ管理］の一覧は B 番号、角度、内 R、直線部 L、親の曲げを示します。BASE は基準板、OFF は抑制中です。抑制は材料を残して直線に戻し、再開で曲げを復元します。削除は選択フランジと、その子のフランジ・成形を削除します。

親番号 0 は基準板です。親変更は共通の展開座標を保ち、つながった一つの板として再構築できる場合だけ適用します。自分・子を親にする循環や分離は拒否します。任意の位置へフランジを付け替える機能ではありません。

線で曲げる ・English

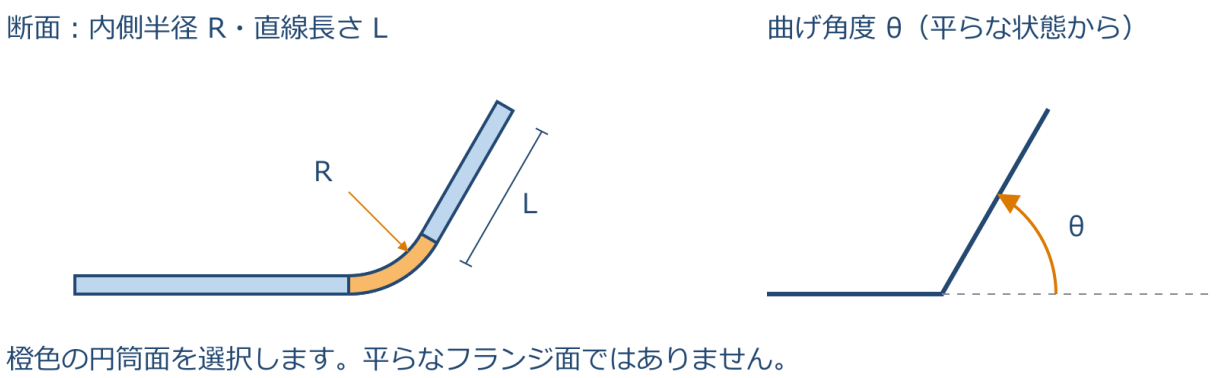


図 49　橙色の円筒面を選択します。平らなフランジ面ではありません。

加工後の再編集・自動 45 度合わせ・寸法基準の使い方

［エッジフランジ］と［線で曲げる］で作った曲げの角度・内側半径を変更します。通常のエッジフランジは直線長さ L も変更できます。

操作の順番

- ［曲げ・フランジ編集］を開き、編集する板金ソリッドを選択します。
- ［曲げ面］で、付け根の丸い円筒面を選択します。内側・外側のどちらの円筒面でも選べます。現在の角度・半径・長さが読み込まれます。
- ［曲げ角度］を変更します。角度は平らな状態からどれだけ曲げるかを表します。必要なら内側半径 R も変更します。
- ［適用］を押します。計算が成功したときだけ形状が置き換わります。［元に戻す］で編集前に戻せます。

線で作った曲げの長さはどうなるか

展開時の板の輪郭と穴位置を保ちます。角度や半径を変えると必要な曲げ帯の長さが変わるため、その残りが直線部分 L になります。L 欄は「適用時に自動計算」と表示され、直接入力はできません。入力中は変更前の長さを表示し、適用後に計算結果へ更新します。

曲げ帯の展開長は 角度（ラジアン）×（内側半径 R＋K 係数×板厚 t）です。K 係数・板厚・曲げの固定側は、この編集では変更しません。

通常のエッジフランジはＬを指定できます。ただし、その平らな部分にさらに線曲げを追加した場合は、元の板と追加した曲げを保つためＬを自動計算します。移動側に付いた曲げも一緒に動きます。

変更できない場合

• 角度は 0 度より大きく 180 度未満、内側半径は正の値で指定してください。

• 別の曲げの付け根を越える場合や、曲げ後に部品が干渉する場合は適用しません。穴や切欠きは展開輪郭の位置を保ちます。

• 展開中は先に［板金再曲げ］してください。通常の面編集で同期された輪郭と、展開後に取り込んだ平面部の貫通穴は保持します。曲げ帯を加工した後も、穴や切欠きを保って角度・Ｒを編集できます。詳しくは通常コマンドでの板金編集を参照してください。

図は各入力欄の意味を示す模式図です。面選択や数値入力だけでは形状を計算せず、［適用］時に計算します。

長さを自動計算する曲げに設定した逃げも、展開形状の切欠きとして保持して角度・Ｒを変更できます。

板金成形：バーリングの使い方

バーリング・ダボ・エンボス・ビード

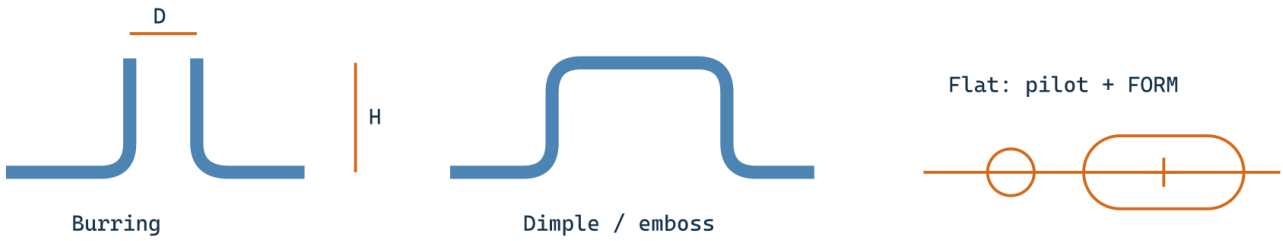


図 50　Form sections and flat representation

- ［板金成形］を開き、ソリッドと成形を置く平面を選択します。
- ［成形の中心位置］を指定し、種類・寸法・配置角度・成形方向を入力します。
- ［適用］します。根元全体が一つの平面内に収まり、穴・曲げ・他の成形と干渉しないことを検査します。

種類	寸法の意味	展開での扱い
バーリング	D＝穴内径、H＝選択面から先端までの高さ、R＝根元の内 R	入力した下穴径の穴を残します。
ダボ	D＝平らな頂部の直径、H＝高さ、根元 R・肩 R	盛り上がりを平板に戻し、成形位置を記録します。
エンボス／ビード	長円形。D＝平らな頂部の幅、直線部長さ＝両半円の中心間、配置角度で向きを指定	平板と、成形輪郭・中心記号を出力します。

内外の面は一定板厚の断面から作ります。肩 R は板厚より大きく、高さは根元 R＋肩 R 以上にしてください。バーリングの高さは根元 R 以上です。［編集］では F 番号を指定して寸法を変更でき、［抑制］［再開］［削除］も使用できます。番号は作成順です。

成形は塑性加工です。完成形状だけから実際の素材の伸び・下穴径を一意に逆算することはできません。下穴径は使用する材料・金型の値を入力してください。展開図の SM_FORM／SM_FORM_CENTER／SM_FORM_NOTE は加工位置の表現であり、切断線 SM_CUT とは別です。

成形から離れた位置の通常の貫通穴・輪郭加工は、曲げ状態と展開状態の両方で保持します。成形自体を通常コマンドで削った形状、根元にかかる穴、曲面上への配置は対象外です。

バーリングは、板に先端が開いた穴の立ち上がりを作る機能です。ダボは頂部が閉じた形、エンボス・ビードは長円形の盛り上がりです。STD／PRO で使用できます。

ソリッド → 基準面 → 成形の中心位置 → 種類と寸法 → 適用

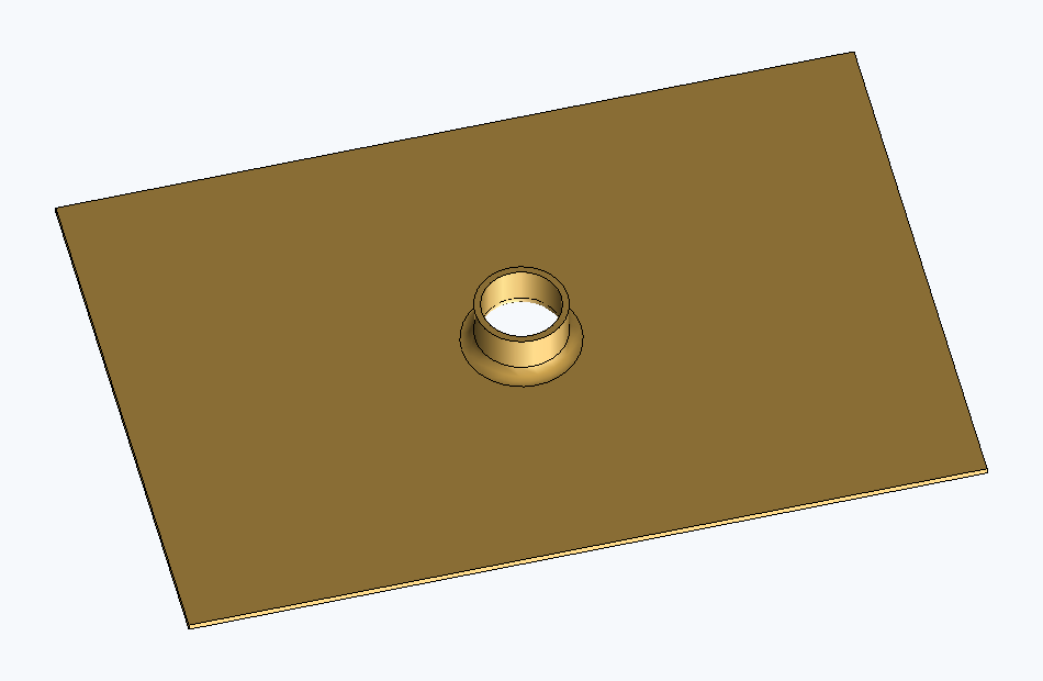


図 51　板と一体のバーリング作成結果

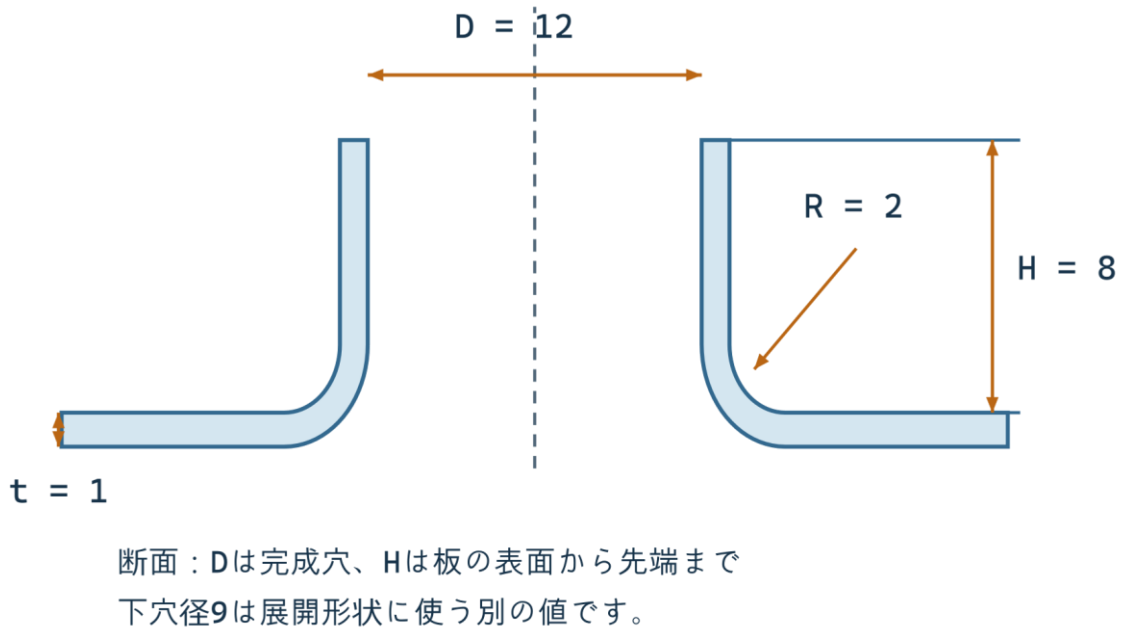


図 52　完成穴径 D、高さ H、根元 R、板厚 t の断面図

1. まずバーリングを 1 個作る

• 板厚 1 mm の基本板を用意します。例 A は 120 × 80 mm です。既存ソリッドなら先に［板金として認識］します。

• ［板金］の［成形］を開き、［成形の操作］を新規作成、［成形の種類］をバーリングにします。

• ［ソリッド］の［CAD で選択］で、板全体を選びます。

• ［基準面］の［CAD で選択］で、板の広い上面を選びます。板厚 1 mm の細い側面は使いません。

• ［成形の中心位置］で、選んだ面の中央を指示します。今回の例では板の左下から X 方向 60 mm、Y 方向 40 mm の位置です。

• ［穴内径／平らな頂部の幅 D］＝12、［成形高さ H］＝8、［根元の内 R］＝2、［展開時の下穴径］＝9 を入力します。

• ［成形の方向］は正方向にします。円形のバーリングでは［配置角度］は 0 度のままで構いません。

• ［適用］で計算・検査します。成功すると板と一体の成形ができ、1 回の［元に戻す］で適用前に戻せます。確認後に［閉じる］を押します。

下穴を先に通常の穴コマンドで開ける必要はありません。成形の根元に既存の穴があると、現在の成形コマンドでは配置できません。穴のない平面から始めてください。寸法の入力中は形状を再計算せず、［適用］で作成します。

2. 寸法と方向の意味

項目	意味
D	バーリングは完成穴の内径。ダボは平らな頂部の直径。エンボス／ビードは平らな頂部の幅。
H	正方向では、選択した板の表面から成形の先端までの高さ。板の裏面からの全高ではありません。
成形の根元 R	平らな板から立ち上がる根元の丸み。バーリングでは H ≥ 根元 R。
肩 R	ダボ／エンボス／ビードの頂部の丸み。肩 R > 板厚、H ≥ 根元 R＋肩 R が必要です。バーリングでは表示されません。
展開時の下穴径	バーリングを展開したときに残す穴径。完成穴 D とは別の入力値です。今回の 9 mm は説明用で、加工条件の推奨値ではありません。
長円の直線部長さ	両端の半円の中心間距離。頂部の全長＝D＋直線部長さ。エンボス／ビードだけに表示されます。
配置角度・成形方向	配置角度は長円の向き。正方向は選択面の外側へ、負方向は板の反対側の面から外側へ成形します。

3. 図面に作成した 6 つの例

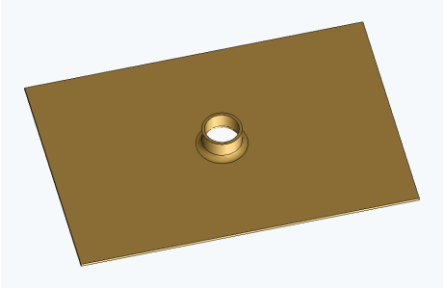
全て 120 × 80 mm、板厚 1 mm。B・D・E・F は正方向、配置角度 0 度です。C は最終位置で B と同寸法を作成してから展開しています。

A・加工前の基本板　　**B・バーリング**　　**C・バーリングの展開**



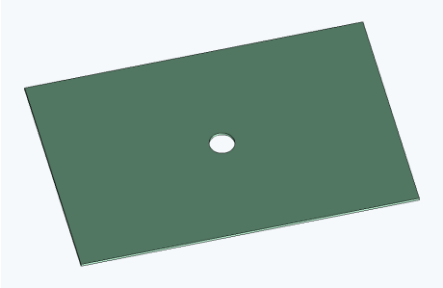
120 × 80、板厚 t=1

この板を使って操作を練習できます。



D=12 / H=8 / 根元 R=2 / 下穴径 9

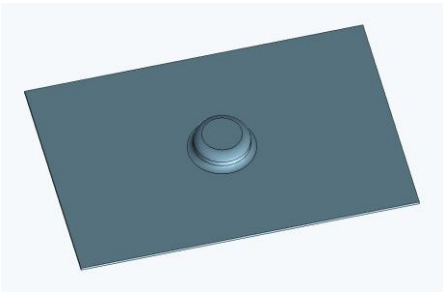
先端が開いた穴の立ち上がりです。



B と同寸法 / 下穴径 9

盛り上がりが消え、指定した下穴が残ります。

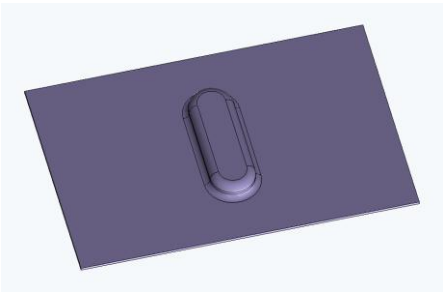
D · ダボ



D=14 / H=7 / 根元 R=2 / 肩 R=3

頂部が閉じた円形の盛り上がりです。

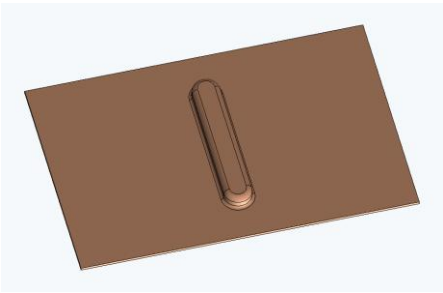
E · エンボス



D=10 / H=6 / R=2 / 肩 R=3 / 直線部 30

長円形の盛り上がり。全長は D＋直線部です。

F · ビード



D=6 / H=5 / R=1 / 肩 R=2 / 直線部 45

細長い例。現在はエンボスと同じ長円断面方式です。

4. 展開・編集・うまくいかないとき

展開と再曲げ	**［板金展開］でバーリングは指定下穴の平板に、他の 3 種類は穴のない平板に戻ります。［板金再曲げ］で成形形状を復元します。**
展開図	［展開図 作成・更新］では、SM_CUT が切断線、SM_FORM／SM_FORM_CENTER／SM_FORM_NOTE が成形輪郭・中心・注記です。成形線を切断線として扱わないでください。
寸法を変える	［成形の操作］＝［寸法を編集］、対象ソリッドと［成形番号（F 番号）］を指定して寸法を変更し［適用］。各例は 1 個なので F1 です。移動・方向変更の入力は新規作成時のみです。
一時抑制・削除	［一時抑制］は平板に戻して成形情報を保持し、［抑制を解除］で戻します。［削除］は成形情報も削除します。
配置できない	成形の根元全体を、穴・板の端・曲げ・他の成形から離れた 1 つの平面に収めます。曲面やロール板、展開状態への新規成形は未対応です。

展開と再曲げ	［板金展開］でバーリングは指定下穴の平板に、他の 3 種類は穴のない平板に戻ります。［板金再曲げ］で成形形状を復元します。
寸法エラー	板厚・D・H・R は正の値を使います。バーリングは H ≥ 根元 R。他の 3 種類は肩 R > 板厚、H ≥ 根元 R＋肩 R を確認します。
モデルの意味	一定板厚の完成形状です。材料の伸び・肉薄化・割れを解析する機能ではありません。実加工用の下穴径は材料と金型の条件で決めてください。

ルーバー・絞り・ブリッジの使い方

板金の［成形］で種類を切り替えて作成します。STD／PRO で使用できます。すべて板と一体の一定板厚ソリッドです。寸法の入力中は再計算せず、［適用］で計算します。

絞りの平板復帰は、製造用の絞りブランク寸法の算定ではありません。ルーバー・ブリッジの切込みは展開図の線で表し、切断幅や塑性変形による伸び代は計算しません。

共通の操作手順

• 120 × 80 mm、板厚 1 mm の［基本板］を作ります。既存ソリッドは先に［板金として認識］します。

• ［板金］→［成形］を開き、［成形の操作］を［新規作成］にします。

• ［ソリッド］→［基準面］→［成形の中心位置］の順に選択します。基準面は広い平面を使います。

• 種類と寸法、配置角度、成形方向を指定し［適用］します。下穴を先に開ける必要はありません。

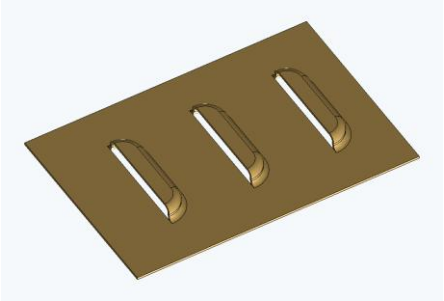
• ［板金展開］で平板に戻し、［板金再曲げ］で成形形状を復元します。切込み線は［展開図 作成・更新］で確認します。

1. ルーバー：片側が開いた通風口

選択位置は切込み線の中央です。D は平らな頂部の奥行き、L は開口の直線部長さ。H は表面からの全高で、開口の高さは H－板厚です。配置角度で開口方向を変えます。0 度は選択面のローカル－X 側に開き、必ずしも画面左ではありません。

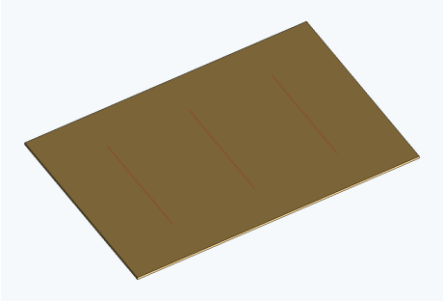
G：板の中心から－35、0、35 mm の 3 か所に作成。H：平板復帰。I：再曲げ。切込み全長は L＋2×(D＋根元 R＋肩 R)、例では 46 mm です。

G · ルーバー / 作成



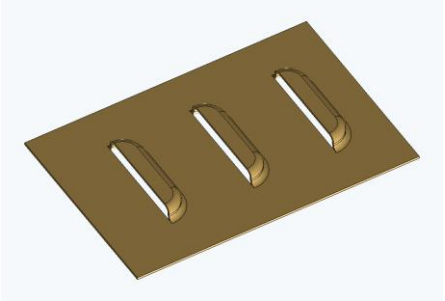
D6 / H6 / 根元 R2 / 肩 R3 / L24

H · ルーバー / 平板復帰



D6 / H6 / 根元 R2 / 肩 R3 / L24

I · ルーバー / 再曲げ



D6 / H6 / 根元 R2 / 肩 R3 / L24

2. 絞り：円形カップと角丸トレー

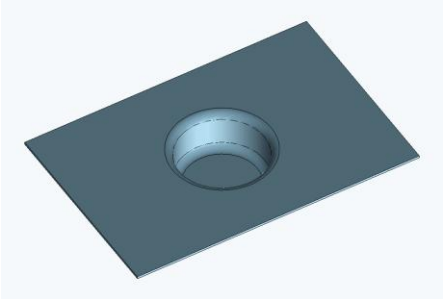
円形絞りは既存ダボと同じ一定板厚の断面生成を使います。負方向にするとカップ状のくぼみになります。角形絞りは角丸長方形の底と 4 側壁を持ち、平面角 R を指定できます。D／W／L は平らな底面の寸法で、開口の外寸ではありません。

J・K・L：円形カップの作成→平板復帰→再曲げ。M・N・O：角丸トレーの同じ手順。いずれも負方向です。

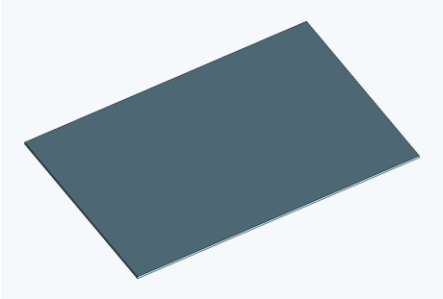
J · 円形絞り / 作成

K · 円形絞り / 平板復帰

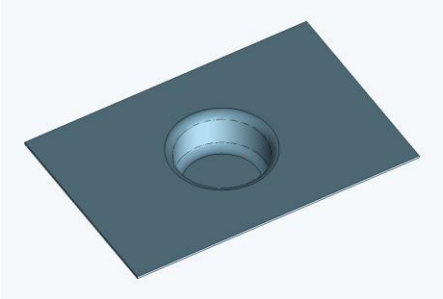
L · 円形絞り / 再曲げ



D30 / H18 / 根元 R2 / 肩 R4 / 負方向

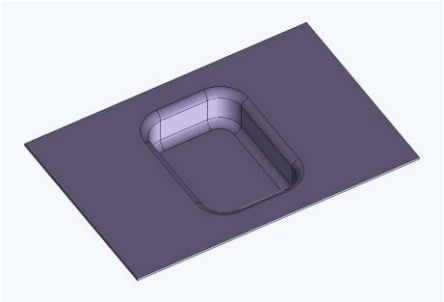


D30 / H18 / 根元 R2 / 肩 R4 / 負方向



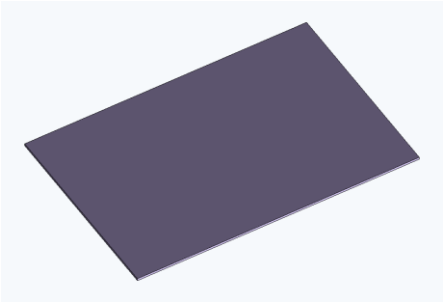
D30 / H18 / 根元 R2 / 肩 R4 / 負方向

M · 角形絞り / 作成



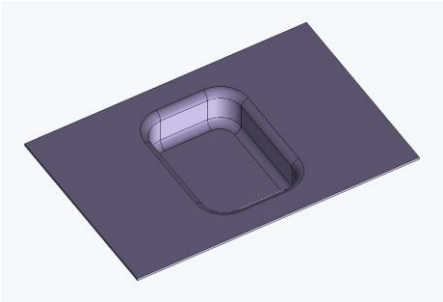
W32 / L48 / H16 / 平面角 R6 / 根元 R2 / 肩 R4 / 負方向

N · 角形絞り / 平板復帰



W32 / L48 / H16 / 平面角 R6 / 根元 R2 / 肩 R4 / 負方向

O · 角形絞り / 再曲げ



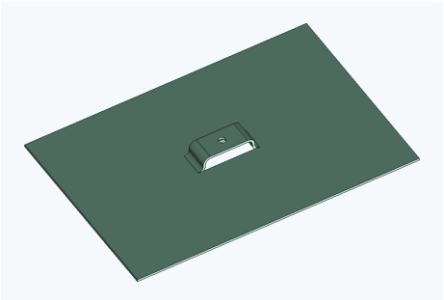
W32 / L48 / H16 / 平面角 R6 / 根元 R2 / 肩 R4 / 負方向

3. ブリッジ：両端をつないだ切り起こし

両側に切込みを入れ、両端をつないだまま中央を持ち上げます。D は平らな頂部の長さ、L は帯の幅、H は表面からの全高。中央穴径 0 で穴なし、正の値で頂部に貫通穴を作ります。中央穴径は D と L の両方より小さくします。

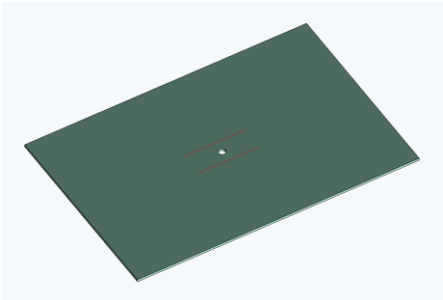
P・Q・R：単発の作成→平板復帰→再曲げ。S・T・U：同じ成形を 46 mm 間隔で 2 か所に配置した 2 連です。2 連専用の種類はなく、位置を変えて 2 回適用します。

P · 単発ブリッジ / 作成



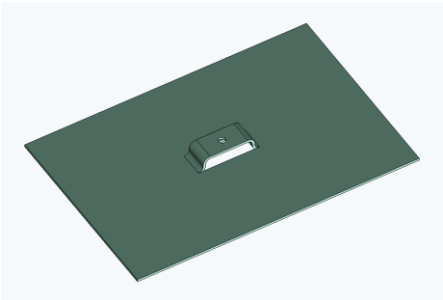
D16 / L10 / H6 / 根元 R2 / 肩 R3 / 中央穴 3

Q · 単発ブリッジ / 平板復帰



D16 / L10 / H6 / 根元 R2 / 肩 R3 / 中央穴 3

R · 単発ブリッジ / 再曲げ

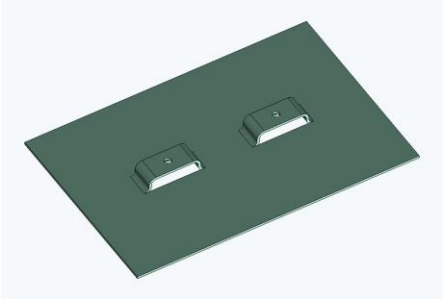


D16 / L10 / H6 / 根元 R2 / 肩 R3 / 中央穴 3

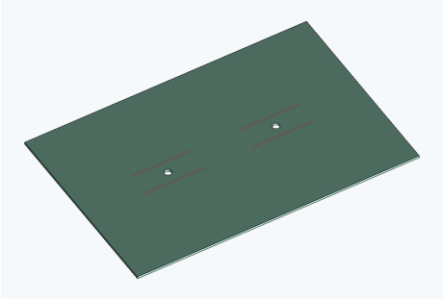
S · 2 連ブリッジ / 作成

T · 2 連ブリッジ / 平板復帰

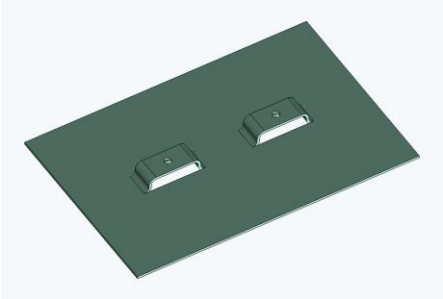
U · 2 連ブリッジ / 再曲げ



単発と同寸法 / 中心間隔 46



単発と同寸法 / 中心間隔 46



単発と同寸法 / 中心間隔 46

寸法・展開・編集の注意点

R と高さ	**根元 R＞0、肩 R＞ 板厚、H ≧ 根元 R+肩 R。角形絞りの幅・長さは平面角 R の 2 倍より大きくします。**
配置範囲	穴・板の端・曲げ・他の成形を避け、必要な平板部分も含めて 1 つの面内に収めます。ルーバーは開口の前方にも平板が必要です。
必要な平板の目安	ルーバー：半径 D＋根元 R＋肩 R＋板厚、直線部 L の長円内。ブリッジ：長さ D＋2×(根元 R＋肩 R＋板厚)、幅 L＋2×板厚の長方形内。
展開時の切込み	ルーバーは 1 本、ブリッジは 2 本のゼロ幅切込みを SM_CUT に出力します。3D 平板に見える幅の溝は作りません。ブリッジの中央穴は 3D 平板にも残ります。
図面のレイヤー	SM_CUT は切断線・切込み線。SM_FORM、SM_FORM_CENTER、SM_FORM_NOTE は成形輪郭・中心・注記です。成形輪郭を切断線として使わないでください。
絞り展開の意味	元の基準平板に復帰します。材料流動、肉薄化、割れ、工程数、製造用ブランク寸法の算定は行いません。金型加工の成立性を保証する機能ではありません。
編集と対象	［寸法を編集］／［一時抑制］／［抑制を解除］／［削除］は F 番号で指定します。新規成形は基本板・曲げた板金の平面部に対応。展開状態やロール曲面には新規配置できません。
更新前の起動中 CAD	追加サンプルは表示できますが、新しい種類のコマンド操作と成形履歴の展開・再曲げには更新版のビルドと次回起動が必要です。既存の起動中 CAD を自動で入れ替えません。

DLL / MCP

sheet.form / sheet.editForm: kind=4 Louver, 5 RoundDraw, 6 RectangularDraw, 7 Bridge. diameter = D/W, span = L, cornerRadius = 角形底面の平面角 R, bridgeHoleDiameter = 中央穴径（0 で穴なし）.

曲げ逃げ：作成・変更・削除

すでに曲げがある板金ソリッドの、曲げ端に逃げを作るコマンドです。逃げがない端への追加だけでなく、このコマンドやフランジ作成で登録された逃げの幅・深さ・形状の変更と削除もできます。平らな板の任意の直線エッジに切り欠きを作るコマンドではありません。

目的	操作	結果
新しく作る	逃げのない曲げ端を選び、長方形／丸形と寸法を指定して適用	その端に逃げを追加
既存の逃げを変更	同じ曲げ端を選び直し、希望する幅・深さ・形状で適用	その端の登録済みの逃げを置き換え
削除	曲げ端を選び、［逃げなし］で適用	その端の登録済みの逃げを解除

1. 選ぶ場所

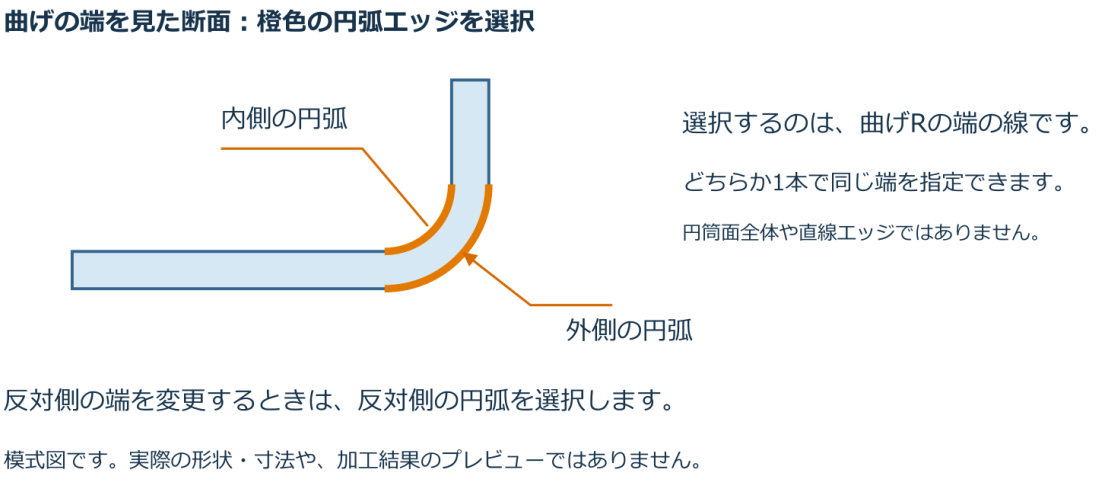


図 74　曲げ端の内側円弧と外側円弧を橙色で表示。どちらか一方の円弧エッジを選ぶ。

円弧は、曲げの円筒面と端の断面が接する線です。既存の逃げで曲げ端が短くなっていても、現在見えている円弧を選びます。丸い切り欠きの奥の円弧とは別です。

2. 新規作成の順番

- ［曲げ逃げ］を開き、［ソリッド］の［CAD で選択］で曲げた板金を選びます。展開中なら先に［板金再曲げ］します。
- ［曲げ端部のエッジ］の［CAD で選択］で、対象の曲げ端の円弧を選びます。選択エッジはオレンジになります。
- ［逃げ形状］を長方形または丸形にし、［逃げ幅 w］［逃げ深さ d］を指定します。
- ［適用］します。選択や数値入力だけでは形状は変わりません。

複数の端を選ぶと同じ条件を適用します。同じエッジをもう一度選ぶと選択を解除できます。適用後はエッジ選択が解除されるため、続けて変更する場合は選び直してください。

3. 作成済みの逃げを編集する

• 対象ソリッドを選び、変更する端の現在の円弧エッジを選び直します。

• 希望する幅 w・深さ d・形状を入力します。長方形から丸形、丸形から長方形への変更も可能です。

• ［適用］します。その端の逃げ設定を置き換え、形状を作り直します。先に削除する必要はありません。

現在、エッジを選んでも既存の逃げの寸法は入力欄へ自動表示されません。表示値を既存形状の測定値と考えず、変更後に使いたい寸法を入力してください。複数端に異なる条件を使う場合は、端ごとに操作します。

4. 幅・深さ・形状の意味

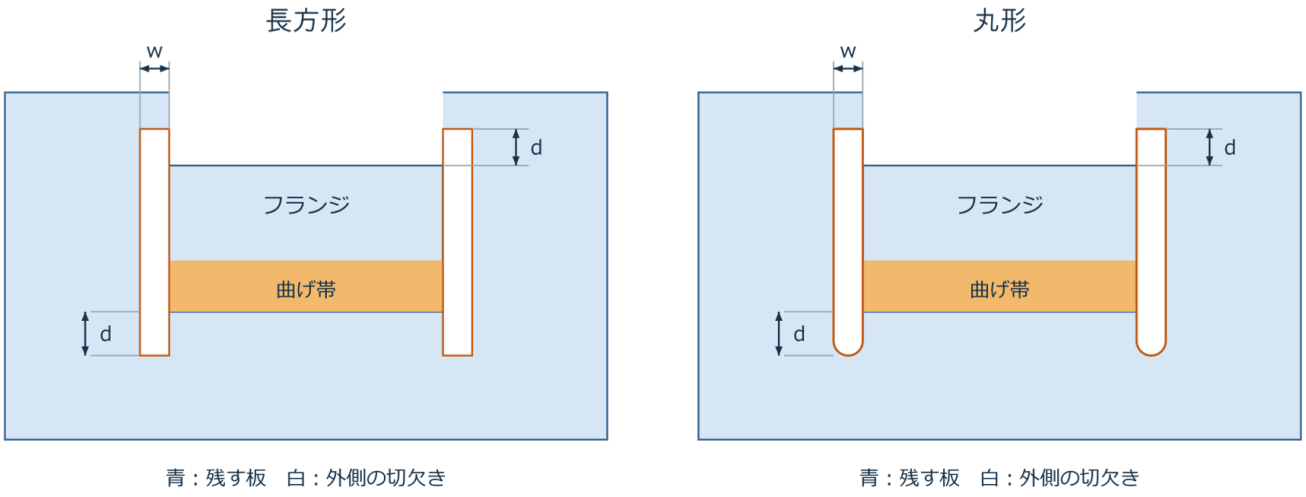


図 75　入隅フランジの展開図。曲げ幅を保ち、隣の板へ幅 w と深さ d の逃げを作る。

ダイアログの上の絵は入隅フランジの模式図です。白い部分が切除される材料で、実物のプレビューではありません。

入力	意味
逃げ幅 w	切り欠きの幅。入隅では曲げ・側板の外側の板に切り込み、曲げと側板自体の幅を保ちます。
逃げ深さ d	曲げ境界から平面部へ入る深さ。入隅の外側方式では基準板へ入る深さで、開口側は展開したフランジ先端より先まで抜きます。
長方形／丸形	角付きの切り欠き／奥を丸めた切り欠き。丸形の入口も板の外へ開きます。
逃げなし	その端に登録された逃げを削除。幅・深さは使いません。値を 0 にして削除する操作ではありません。

自由端や線で作った曲げなどでは、曲げ帯の端と両側の平面部へ切り込む方式になります。入隅の外側方式と切除範囲が異なります。板金設定の形状図も参照してください。

5. 編集できる逃げ・できない切り欠き

• 対象：このコマンドで作った逃げ、フランジ作成時に自動登録された逃げ。

• 別の操作：箱の板金変換で作られた角の隙間や根元の開口、通常の切除・面移動で作った切り欠きを、このコマンドの w・d として自動認識することはありません。赤・黄色の端を個別に調整する場合は通常の面移動・オフセットを使います。

• 通常編集した後：登録された逃げが通常加工の形状に取り込まれた場合は、このコマンドで元の逃げ寸法を取り出して変更・削除することはできません。曲げ帯に穴などを加工した形状では、この板金専用編集を受け付けない場合もあります。

6. 値を変えても形が変わらない場合

まず［適用］後のダイアログ下部のメッセージを確認してください。条件に合わない場合は、元の形状を残したまま適用を中止します。数値や形状を選び直しただけでは、実物は変わりません。

選択箇所：曲げに沿った長い直線ではなく、左右の端にある短い円弧を選びます。対象外の直線は選択できません。丸形：d > w/2 が必要です。幅を大きくした場合は、深さも確認してください。長方形にはこの丸形専用の条件はありません。

w・d は正の値にします。現在の実装では、w は曲げ幅の半分未満、d はフランジの直線長より小さくし、丸形では d > w/2 が必要です。条件内でも材料が分離・干渉する形状は作成しません。失敗した場合は元のソリッドを保持します。

逃げを削除して材料を戻す場合にも、周囲との接続・干渉を確認します。通常加工で作った穴や切り欠きをすべて埋めるコマンドではありません。適用後は［元に戻す／やり直し］が使えます。

切れ目作成：箱の角と、面上のスリット

板金設定 ・English

従来の外周切除に、箱の角を開く切れ目と面上の直線に沿う貫通スリットを追加しました。通常のソリッド・読み込んだソリッドと、JDesignCAD の板金ソリッドで使えます。操作はモデル空間で行います。

どの種類を選ぶか

切れ目の方法	選択するもの	用途
箱の角（エッジ）	ソリッド＋縦の直線角エッジ 1 本	直角な 2 つの壁の共有する角を開く。内側・外側どちらの角も選べます。
箱の角（2 面）	ソリッド＋角を挟む平らな壁面 2 面	共有エッジが存在しない板金フランジにも使用。2 面の選択順で第 1 側・第 2 側を指定。
面上の線からスリット	ソリッド＋平面＋その面上に描いた直線	板の途中の貫通スリット、または端からの切込み。曲げ済み形状の平坦な壁にも使用。
外周エッジの切除（従来）	板金の平板・展開形状＋外周の直線エッジ	板の端を帯状に削る。単純な平板の結果は端面を内側へ移動する操作に近いものです。

箱の角を開く

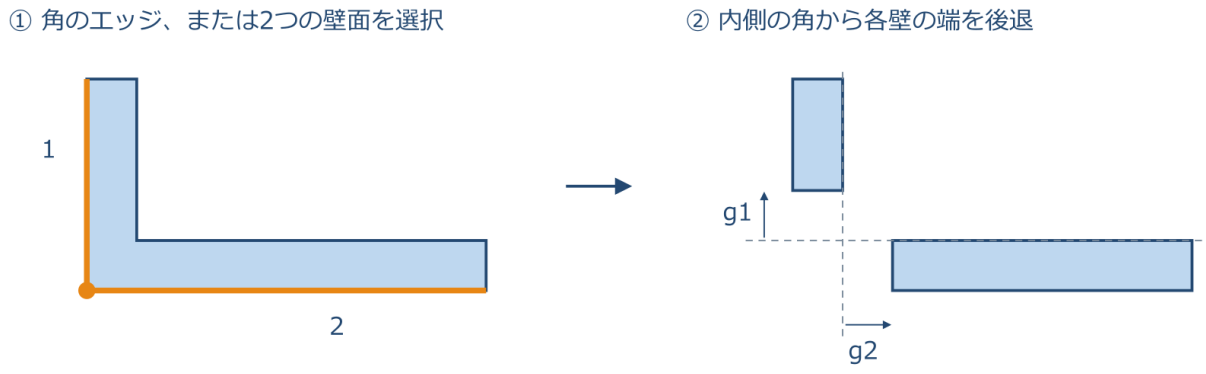


図 76　上から見た断面。破線：内側の角　g1・g2：後退量　両側均等なら g1 = g2 = g/2。

- ［切れ目作成］で［箱の角（エッジ)］または［箱の角（2 面)］を選びます。
- ［ソリッド］で箱を選びます。
- エッジ方式なら縦の角を 1 本選びます。2 面方式なら角を挟む広い壁面を 2 回のクリックで選びます。両方とも内面、または両方とも外面にします。板厚の細い端面は選びません。
- ［隙間幅］g と［隙間の方向］を指定します。最初は［両側均等］が分かりやすい設定です。
- ［適用］します。底板と曲げ帯を残し、壁の平坦な範囲に切れ目を入れます。

g は各壁の端の後退量に割り当てる値です。第 1 側なら g1=g・g2=0、第 2 側なら g1=0・g2=g、両側均等なら g1=g2=g/2 です。図の内側の角を基準にします。斜め方向の最短距離が g になる意味ではありません。既に開いている角へ適用しても材料を付け足しません。現在より狭い後退量なら変更できない場合があります。

エッジ方式の第 1 側・第 2 側は内部の面順になります。片側を指定する場合は 2 面方式で選択順を決めてください。曲面の角、直角でない角、丸い曲げ帯を横切る切除には対応していません。

板の途中にスリットを入れる

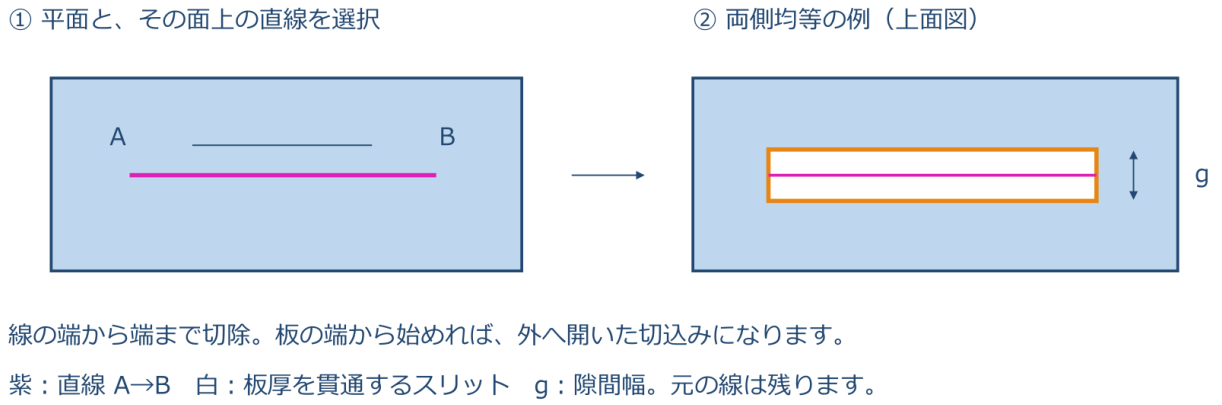


図 77　紫：直線 A→B　白：板厚を貫通するスリット　g：隙間幅。元の線は残ります。

- 通常の線分コマンドで、対象の平面上に始点 A から終点 B まで直線を描きます。
- ［切れ目作成］で［面上の線からスリット］を選びます。
- ［ソリッド］→［スリットを入れる面］→［スリット線］の順で選びます。
- 幅 g と方向を指定します。選択面を外側から見て A→B に進む右側が第 1 側、左側が第 2 側です。両側均等は線の両側へ g/2 ずつです。
- ［適用］すると線の長さに沿って板厚を貫通します。直線自体は残ります。

線を自動延長せず、端から端までを切除します。スリット端は直角です。線だけでなく指定幅の全体が同じ平面の材料内に収まるようにしてください。穴・曲げ帯を横切るもの、複数ソリッドに分離する切除、丸端・曲線のスリットは対象外です。

板金情報と通常ソリッドの違い

対象	加工後
JDesignCAD の板金情報を持つ形状	平坦なパネルの切除を展開座標で保持し、展開・再曲げに引き継ぎます。底の曲げ帯の形は保持します。
通常／読み込んだソリッド	角の切れ目とスリットを作成できます。切れ目を入れるだけで、既存の曲げを認識して展開できるようになるわけではありません。読み込んだ曲げ形状の自動認識は別機能です。

板金情報付きの曲げ状態を通常コマンドで変更してある場合、その変更を展開情報へ自動では取り込めません。適用時に整合性を確認します。通常加工を挟む場合は、展開して加工し、再曲げする手順を使います。

従来の外周切除

第 1 側は板の内側へ g、第 2 側は外側、両側均等は内側へ g/2 を切除します。外側には材料がないため第 2 側では実用上切除されません。平板または展開状態の板金で、自動判定した基準面の外周エッジを選びます。選べない場合は表裏の反対側のエッジも確認してください。曲げ帯を横切ると再曲げできません。

表示とやり直し

ソリッドは緑の輪郭、対象面・エッジはオレンジの輪郭、スリット線は紫です。面を塗りつぶさず、線を見やすくしています。選択・入力では形状を変更せず、［適用］で計算します。［元に戻す］・［やり直し］に対応します。上部の図は説明用の模式図です。

他 CAD の調査（2026 年 10 月 9 日）

CAD／公式資料	機能
Autodesk Fusion	エッジに沿う切れ目、面上の 2 点間の切れ目、面全体の切除。
Autodesk Inventor	単一点・2 点間・面の範囲を使う方式。板面に垂直な切除と、隙間の側を選択。
Onshape	2 つのエッジ／面から結合を定義。曲げまたは切れ目、エッジ接合・方向別の突合せ接合、モデルの最小隙間。

今回の対応は直角の平面壁の角を開く操作と、描いた直線に沿うスリットです。面全体の削除、直接 2 点を指示する入力、接合の関連付けや突合せ形状への延長、任意の曲面を横切る処理までは含みません。

箱・L 形・コの字を、展開できる板金に変換する

「板金として認識」の変換方法：箱・L 形・コの字を板金へ変換を使います。底板と側板が直角に接続した通常のソリッドを、側板がある辺に曲げ R を持つ板金へ作り替えます。箱の位置・外寸・実際の板厚を保ちます。形状が変わるため、切れ目作成だけとは異なります。

対象は長方形の底と、同じ高さで直角に立つ側板 1〜4 枚がある、一定板厚のソリッドです。側板 3 枚、向かい合う 2 枚、隣り合う 2 枚、1 枚の L 形にも対応します。角の縦切れ目があっても使えます。穴・段差・既存の曲げ R があるものは対象外です。通常コマンドで作った箱でも、読み込んだ箱でも、同じ形状条件で検査します。

操作の順番

• 変換前の箱を開き、［板金として認識］を実行します。

• ［変換方法］を［箱・L 形・コの字を板金へ変換］にします。

• ［ソリッド］で箱全体を選びます。箱は緑の輪郭で表示します。

• ［固定する底面］で、箱の内側の底、または外側の底の広い面を選びます。選んだ底面はオレンジで表示します。側壁や上端の細い面は選びません。

• 通常は［板厚の指定］を［自動検出］にし、内側曲げ半径 R・K 係数・角の逃げ幅 g を設定します。

• ［適用］を押します。ここで板厚と箱の形を検査し、側板がある辺だけに 90 度の曲げを作り、側板どうしが接する角に切れ目を作成します。

• ［閉じる］で終了し、図面を保存します。［板金展開］で箱を開けます。

側板どうしの角に必要な切れ目は自動作成します。先に［切れ目作成］を使う必要はありません。板金 11 のように底板の上面から側板の上端まで角を切り離した形も対象です。既存の隙間を狭めず、指定した逃げ幅が大きい場合に広げます。板の途中のスリットや、底板まで切り込んだ形は対象外です。

変換でできる形

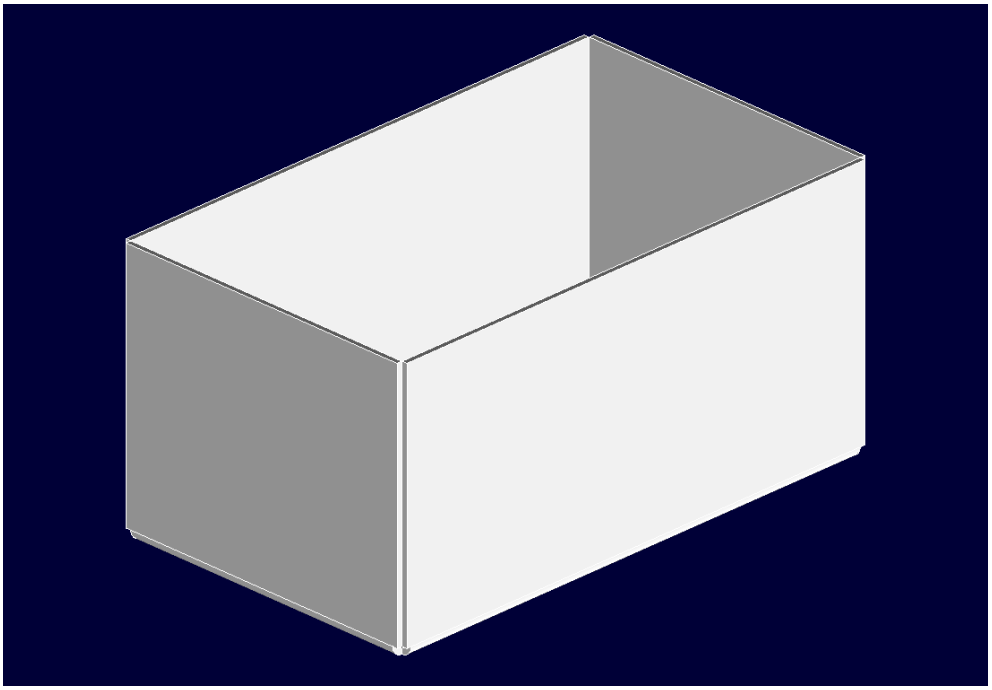


図 78　板金 10 を変換した箱。縦の角に隙間があり、底と側壁は曲げでつながる。

残っている側板の外側の面と上端の位置を保ちます。側板のない辺には曲げを作りません。底の内側には R、外側には R＋t の曲げを作り、曲げ端部が重ならないように根元を開きます。元の鋭い角の形状は変わります。曲げには共通の R と K 係数を使います。隣に側板がない端は元の幅を保ち、角の逃げ幅 g では短くしません。

側板のパターンと展開形状

板金 12：側板 3 枚 → 曲げ 3 か所

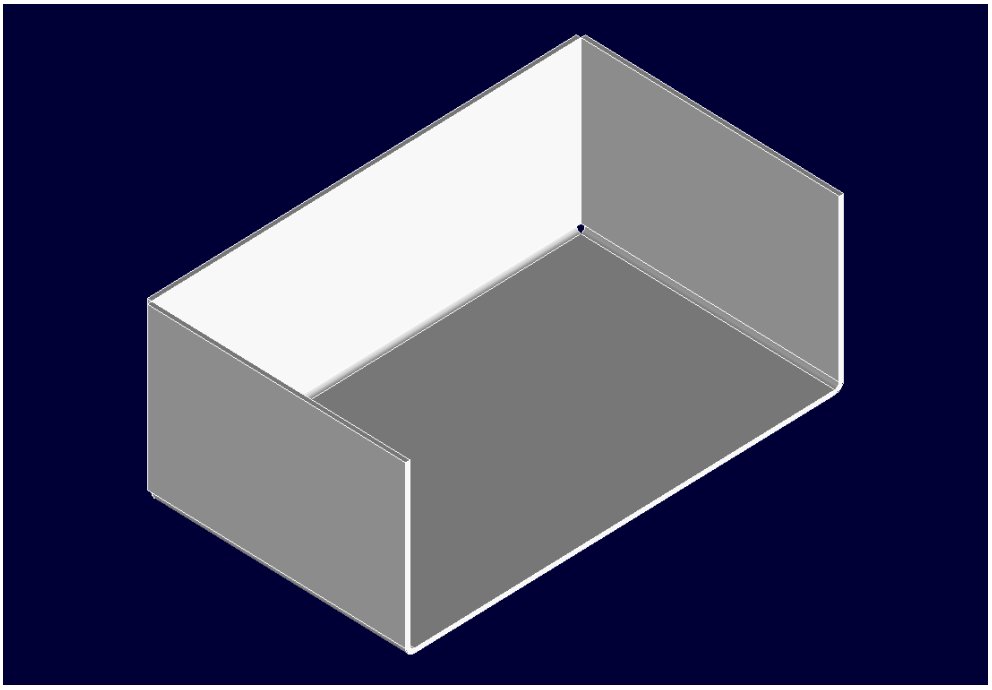


図 79　板金 12 の変換後

変換後



図 80　板金 12 の展開後

展開後

板金 13：向かい合う側板 2 枚 → 曲げ 2 か所

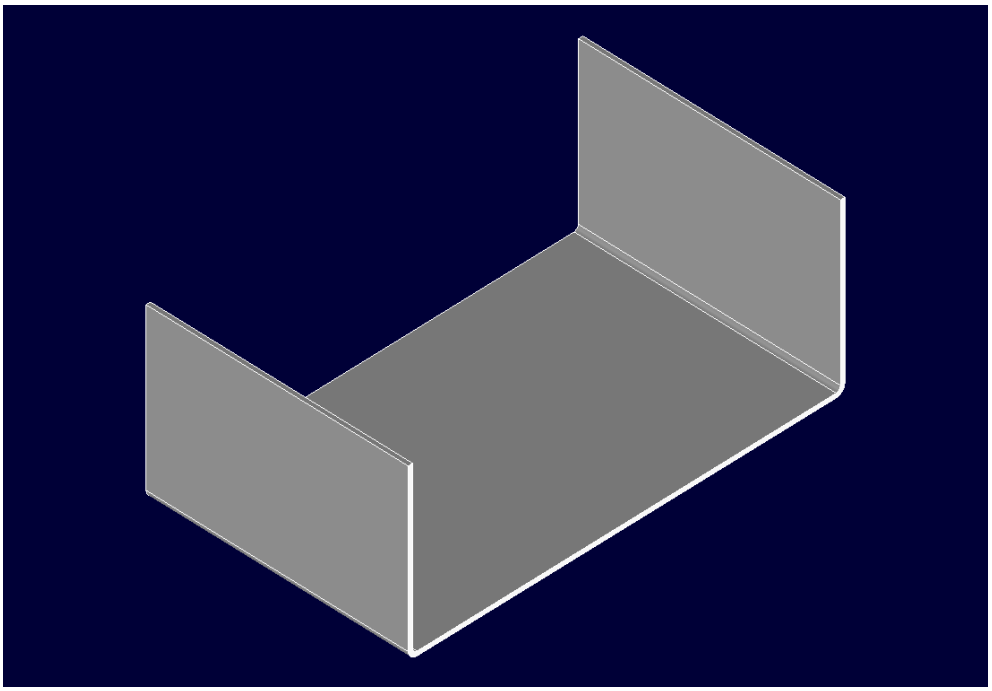


図 81　板金 13 の変換後

変換後



図 82　板金 13 の展開後

展開後

板金 15：隣り合う側板 2 枚 → 曲げ 2 か所

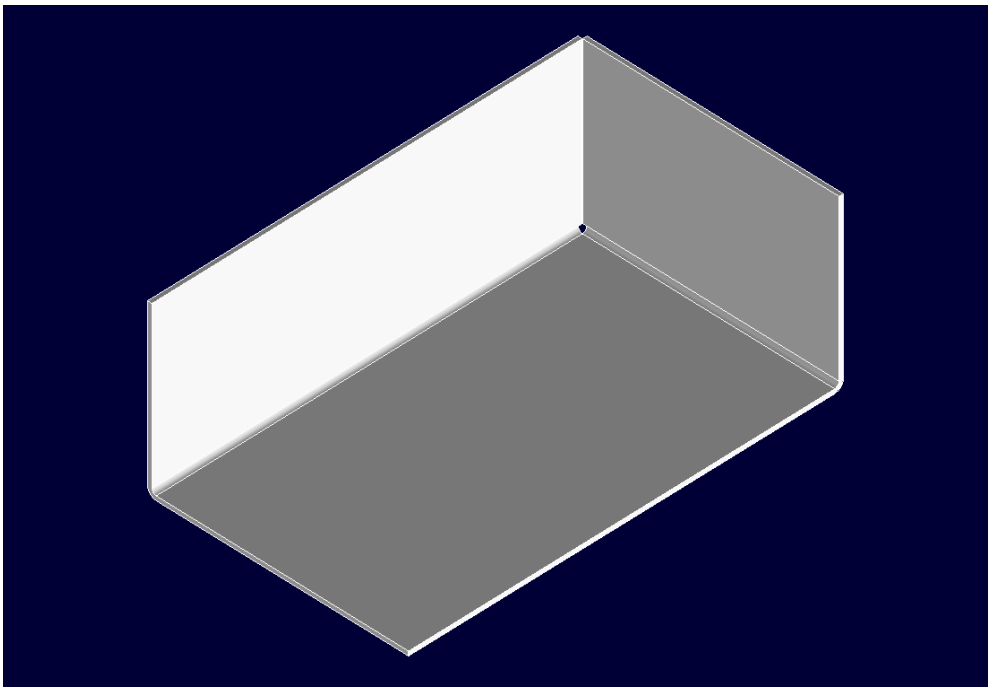


図 83　板金 15 の変換後

変換後



図 84　板金 15 の展開後

展開後

板金 16：側板 1 枚・L 形 → 曲げ 1 か所

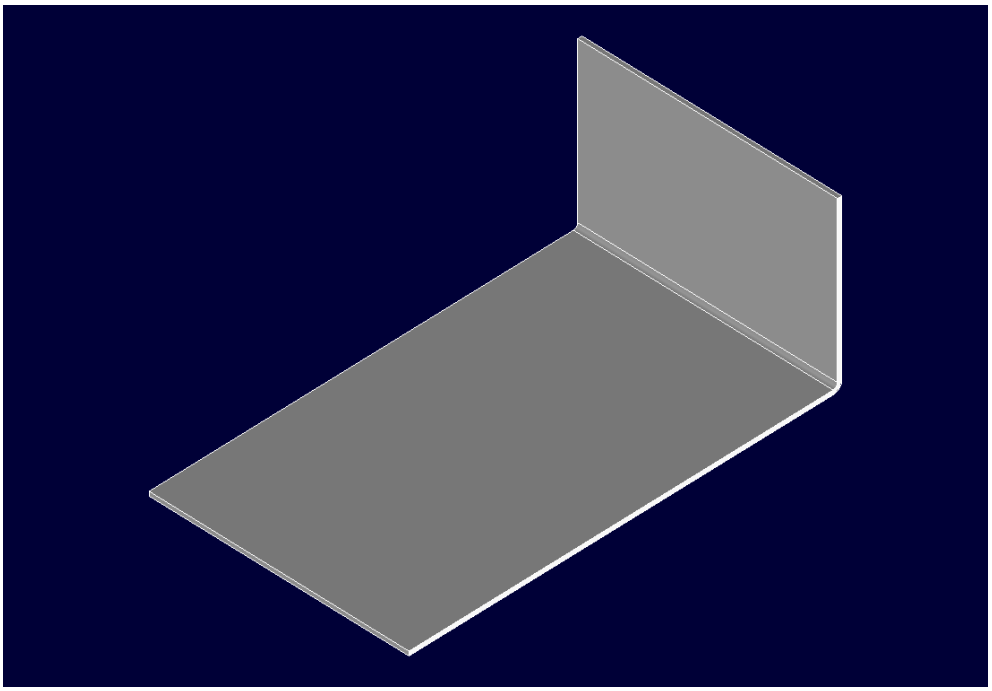


図 85　板金 16 の変換後

変換後



図 86　板金 16 の展開後

展開後

入力値が示す場所

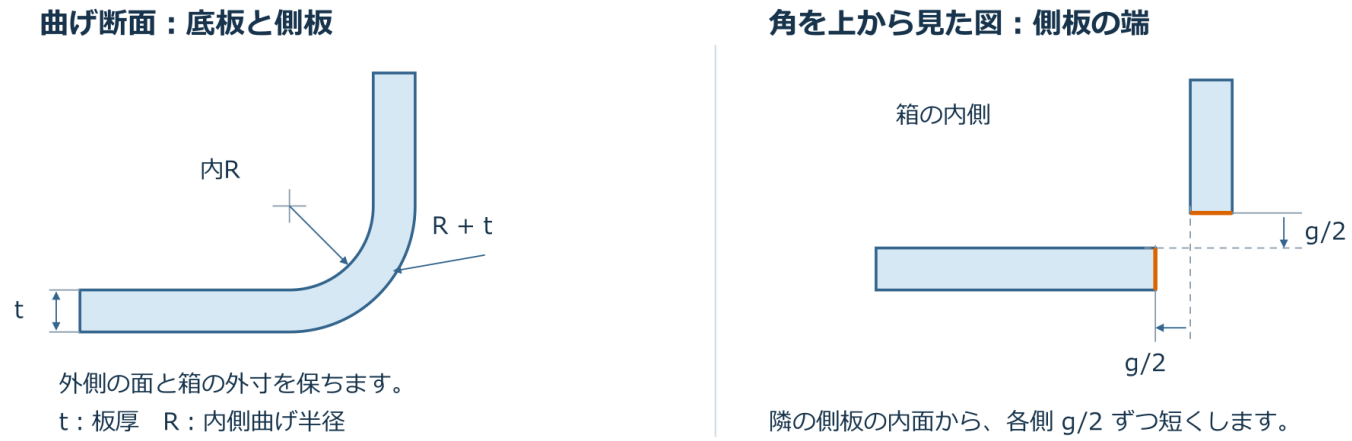


図 87　左は底と側板の曲げ断面の板厚 t と内 R。右は角を上から見た図で、隣の壁の内面から各側 g/2 ずつ短くする。

項目	意味
板厚 t	底板と各側板の厚さが同じであることを検査します。［自動検出］のグレー欄は、適用前には測定結果ではありません。［数値を指定］では実際の板厚と一致することを確認し、箱の厚さ自体は変えません。
内側曲げ半径 R	底板と側板をつなぐ内側の丸み。外側半径は R＋t です。側板がある辺では R＋t だけ底板の平面部を内側へ寄せるため、残る底板の幅・奥行が正である必要があります。R＋t は側板の高さより小さくします。
K 係数	展開長の計算条件です。各 90 度曲げの曲げ帯の展開長は (π/2)×(R＋K×t) です。

項目	意味
角の逃げ幅 g	隣の側板がある角で、その内側面を基準に、両側の側板端をそれぞれ g/2 ずつ短くします。既存の切れ目がこれより大きければ保持します。g は斜め方向の最短距離ではありません。隣接する側板がある形では、0 より大きく、2R 以下で指定します。側板 1 枚や向かい合う 2 枚だけの場合、g は形状に影響しません。曲げ根元の開口は R・t・g から自動で決まります。

このモードの角の逃げ幅 g は、［曲げ逃げ］の幅 w・深さ d とは別の項目です。ここでは側板どうしの角に共通の条件を使い、丸形／長方形の逃げ指定は使いません。加工条件が変わる場合は、元の箱から変換し直してください。

展開して加工し、再曲げする

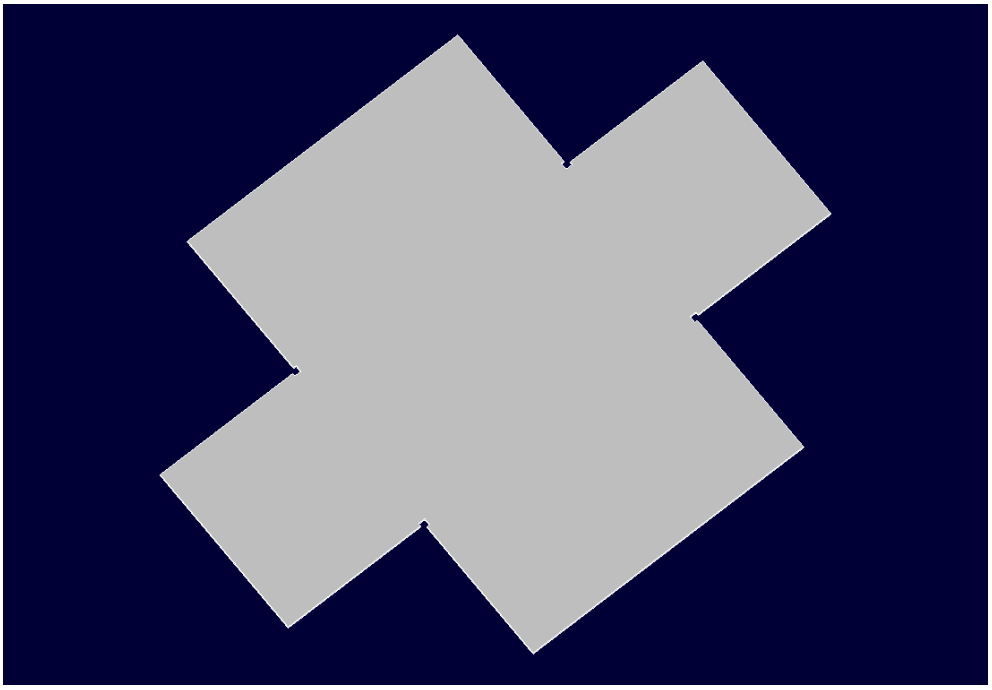


図 88　変換した箱を展開した、底板と 4 側板が一体の十字形の板。

［板金展開］で登録した側板が開き、同じモデル空間に 1 枚の板金ソリッドができます。通常コマンドで曲げをまたぐ貫通穴・長穴や輪郭の編集を行った後、［板金再曲げ］で曲げ状態に戻せます。展開ソリッドには通常、曲げ境界の表示線はありません。曲げ線付きの図面が必要なら［展開図作成・更新］を使います。

曲げ帯をまたぐ貫通加工と、一定板厚を保った輪郭の伸縮に対応します。止まり穴、板厚変更、材料が分離・干渉する編集は対象外です。元の外寸に合わせて変換した箱でも、その後に曲げ角度や R を編集すると外寸は変わります。

うまくいかないとき

• 「底の広い面」を求められたら、側壁や上端ではなく底面を選び直します。

• 箱形状の対象外と表示されたら、長方形の底と側板だけの、穴や丸みのない形を使用します。板厚違い・高さ違い・斜めの側板・辺の途中だけにある側板も対象外です。

- 変換済みの部品は、変換を再適用せず［板金展開］を使います。
- 変換に失敗した場合、元のソリッドは置き換えません。成功後は［元に戻す］で元の箱へ戻せます。

変換後に側板と曲げの幅を変更したい場合

通常の［面の移動］［面のオフセット］で、側板の細い端面または曲げ端の断面を選び、幅方向へ移動します。選んだ端面だけが伸縮し、もう一方の端と底板の平面部は固定します。赤・黄色の両方を選ぶと同じ量で動き、元の位置差を保ちます。展開状態の輪郭編集や、板厚を保つ輪郭テーパーにも対応します。

図入り手順：幅編集・曲げをまたぐ長穴・台形フランジ

板金コーナー処理：選ぶ面と隙間の意味

隣り合うフランジの端を調整し、角の合わせ方や隙間を整えるコマンドです。接合方法によって、選ぶ面と「隙間幅 g」の意味が変わります。

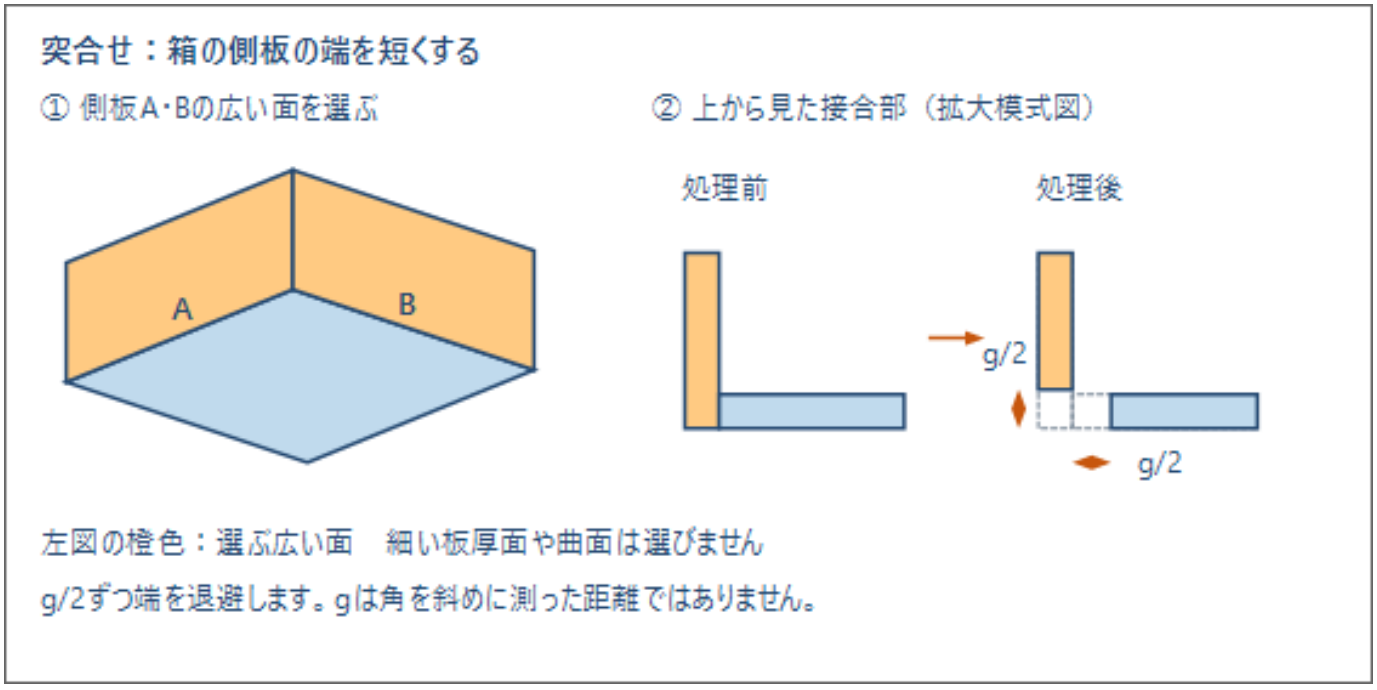


図 89　突合せ：側板の広い面 A と B を選び、各端を g/2 退避する図

突合せ：箱の側板の端を短くする。

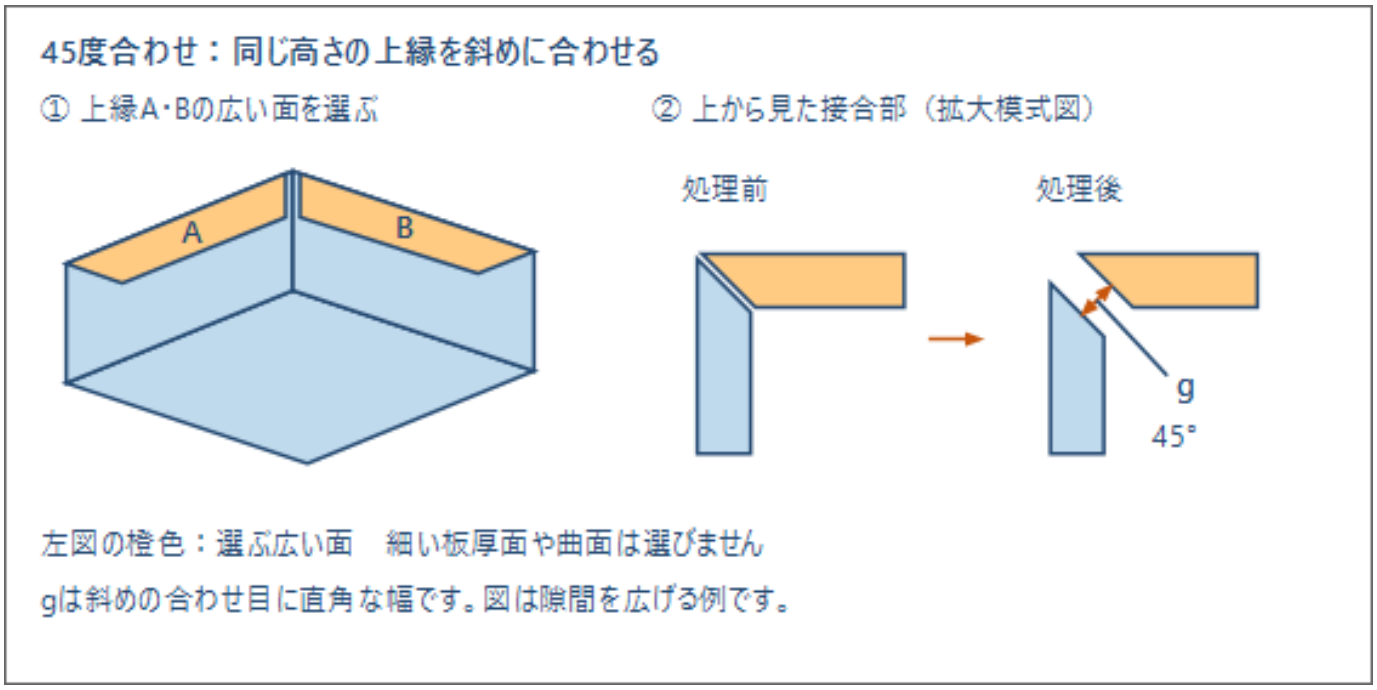


図 90　45 度合わせ：同じ平面の上縁 A と B を選び、斜めの合わせ目に隙間 g を作る図

45 度合わせ：同じ高さの上縁を斜めに合わせる。

最初に選び分ける

接合方法	選択する面	隙間幅 g
突合せ	基準板の角を共有する、側板の広い面を 2 枚。1 つの側板の表裏 2 面を選ぶのではありません。	両端の退避量の合計。各端を g/2 ずつ退避します。
45 度合わせ（同一平面）	同じ平面にあり、隣り合う直交フランジの広い面を 2 枚以上。箱の上縁 4 面もまとめて選べます。	斜めの合わせ目に直角に測った幅です。

細い板厚面や曲げの円筒面は選びません。角を丸めるフィレット、曲げ逃げ、板同士の溶接・合成とは別の操作です。

A：突合せの例

基本板 80×50 mm、板厚 2 mm、内 R3 mm、側板の直線部 20 mm、曲げ 90 度。隣接する 2 枚の側板を選び、g＝4 mm を指定します。

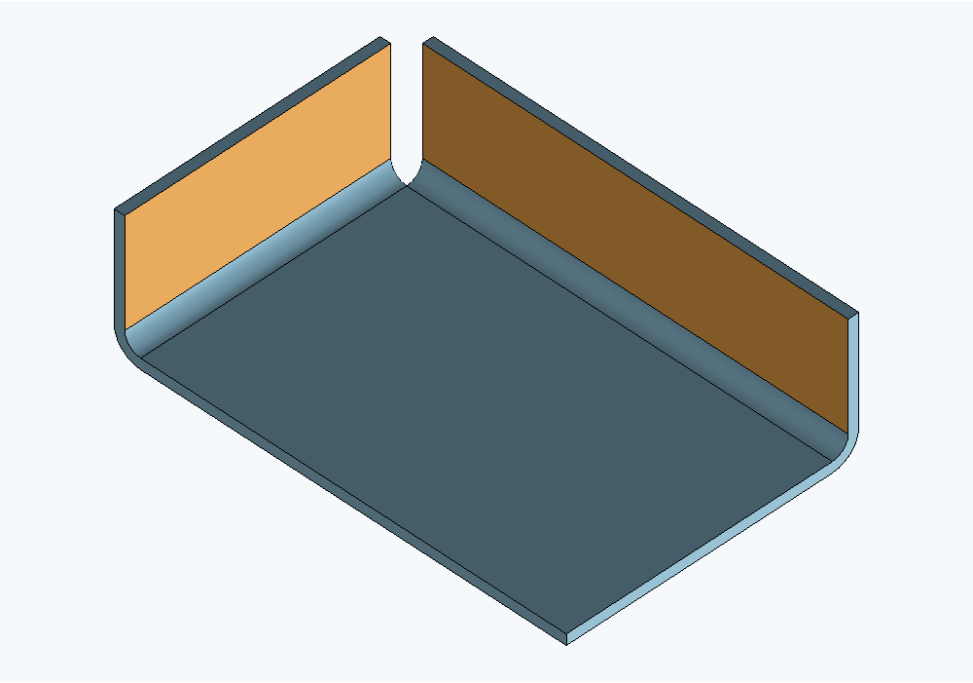


図 91　突合せの処理前。選ぶ側板を橙色で表示

A1 処理前。橙色の 2 枚の側板から広い面を 1 面ずつ選びます。

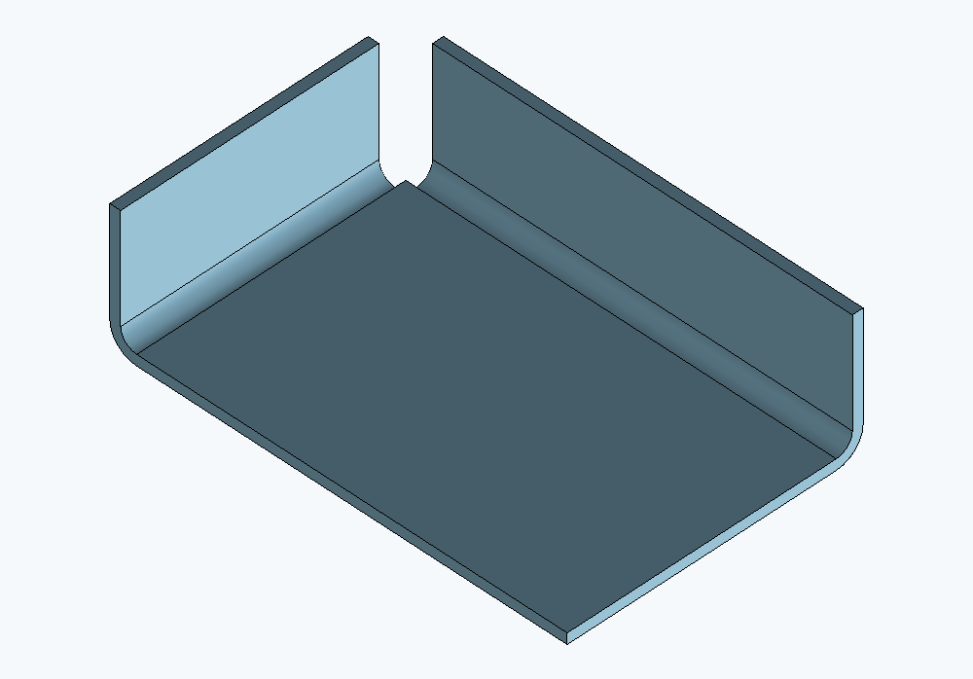


図 92　突合せで端をそれぞれ 2mm 退避した形状

A2 処理後。角に近い端を各 2 mm 退避します。

g＝4 mm は、角を斜めに測った最短距離が 4 mm になるという意味ではありません。曲げ R や板厚による元からの開口と、追加する端の退避を区別してください。

B：45 度合わせの例

箱の外形 100×60 mm、板厚 1 mm、内 R1 mm、上縁フランジの直線部 20 mm。処理前はすでに 0.2 mm の隙間で斜めに合わせてあります。上縁 4 面を選び、g＝2 mm へ変更します。

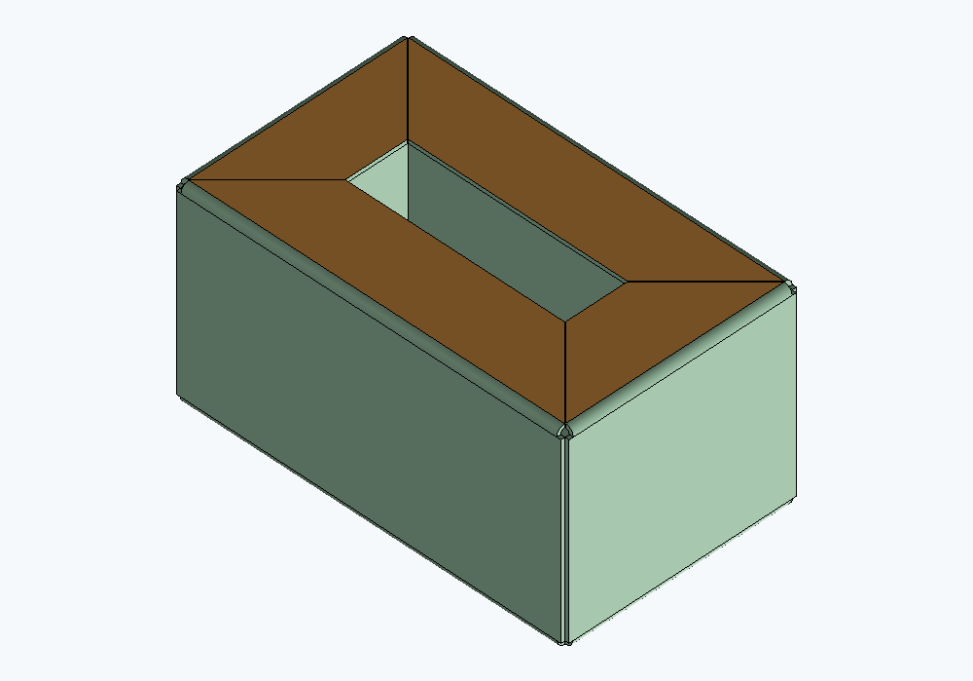


図 93　45 度合わせ変更前。上縁 4 面が橙色で、合わせ目 0.2mm

B1 変更前。橙色の上縁 4 面を選びます。底面や縦の側板は選びません。

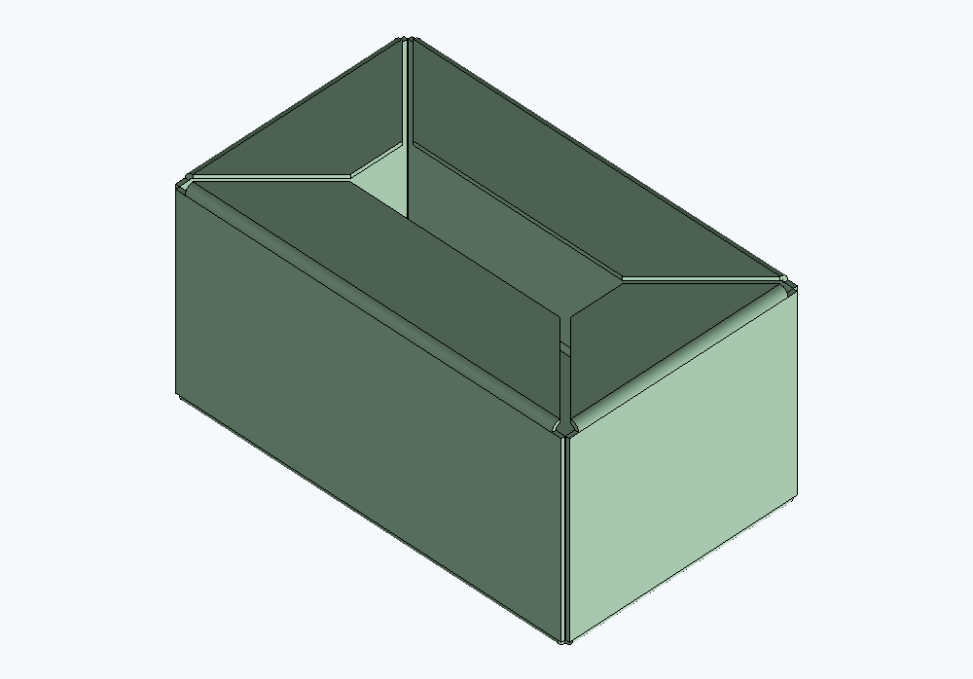


図 94　45 度合わせ変更後。4 角の合わせ目幅 2mm

B2 変更後。4 角の斜めの合わせ目が 2 mm になります。

内向き・外向きに対応し、隙間を広げる・狭める両方が可能です。任意角度、段違いの面、重ね継ぎは対象外です。個別に輪郭加工したフランジや、曲げ幅を削る旧形式の逃げがある形状は使えない場合があります。範囲指定で親板側に作った逃げは併用できます。

ダイアログでの手順

- ［板金］→［コーナー処理］を開き、［接合方法］を選びます。
- ［ソリッド］の［CAD で選択］で板金部品を選びます。
- ［隣接フランジ面］の［CAD で選択］で、上の図の広い面を選びます。突合せは 2 枚、45 度合わせは 2 枚以上です。
- ［隙間幅］を入力し、［適用］します。計算は適用時に行います。
- 必要に応じて［板金展開］→［板金再曲げ］を使います。

新しい説明図は接合方法に連動します。数値入力だけでは図の形は変わりません。図は寸法位置を示す模式図です。

同梱サンプルの構成

上段 A が突合せ、下段 B が 45 度合わせです。それぞれ左から 1 処理前 → 2 処理後 → 3 展開 → 4 再曲げを並べています。8 個のソリッドと 20 個の説明文字で構成し、板金情報を保持しています。

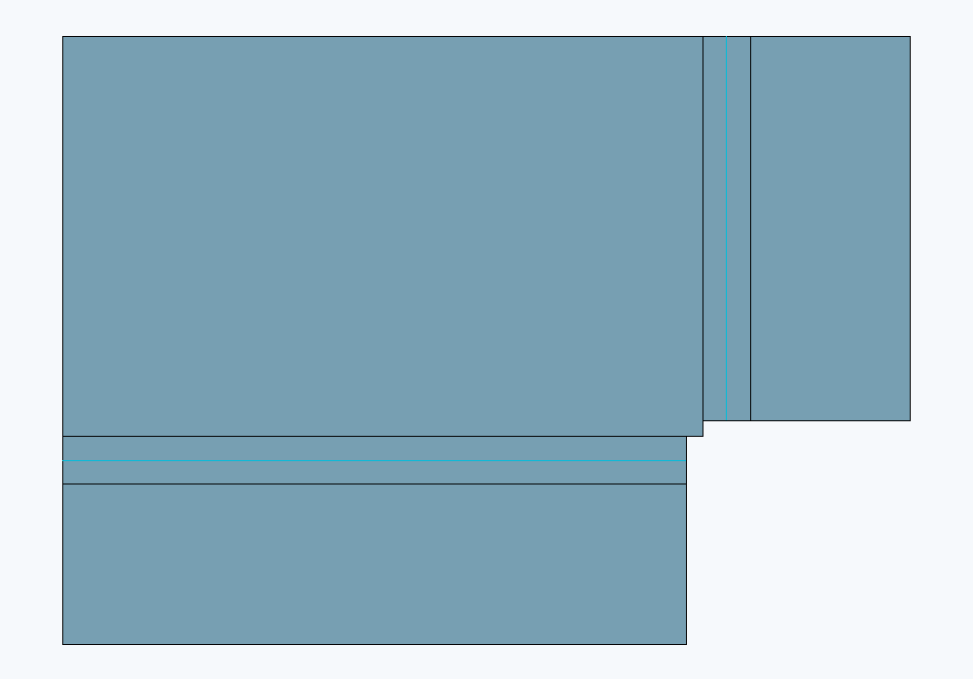


図 95　突合せ処理後の展開形状

A3 突合せ後の展開。

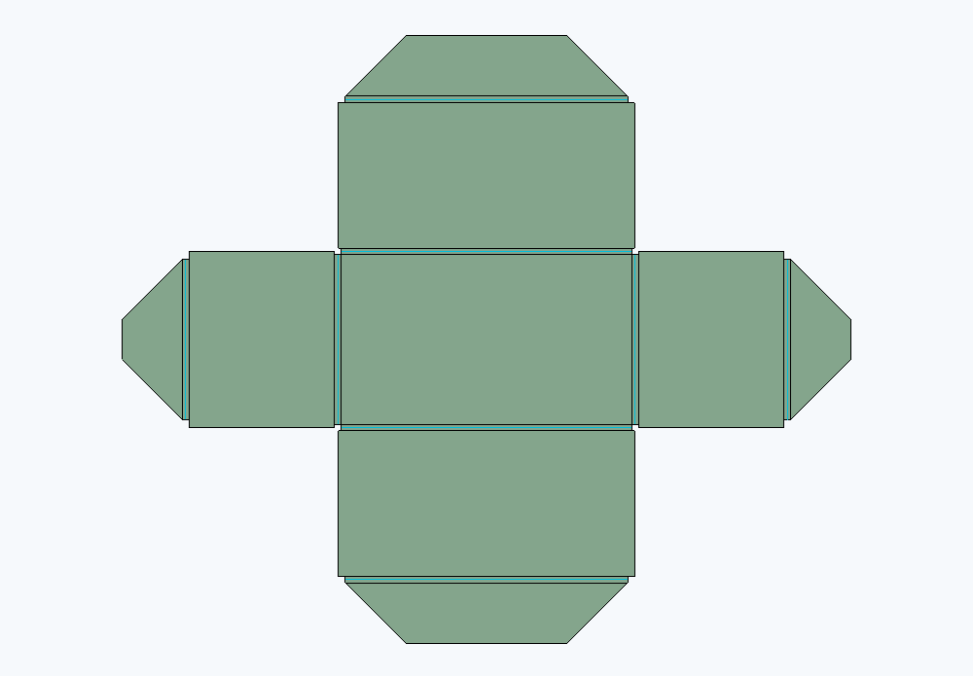


図 96　45 度合わせ後の箱の展開形状

B3 45 度合わせ後の展開。

サンプル JDC を開く

この例では、ソリッドの妥当性、突合せの退避量、45 度合わせの実際の隙間幅、展開後の再曲げ形状の一致を確認しました。図示用の寸法例であり、製造条件の推奨値ではありません。

展開補正表：CSV の書式と「伸び値」の意味

既存部品の板厚と展開補正表

［板金設定］の対象を［選択した板金］にし、板厚を変更します。展開の実輪郭と貫通穴を保持し、曲げ帯を再配分します。部分展開中は先に再曲げしてください。新しい板厚で干渉する場合は変更しません。

［展開補正表］では、プロジェクト既定値または選択部品へ CSV を読み込みます。列は次の順で、寸法単位は mm、角度は度、数値の小数点は「.」です。

```
Material,Tool,Thickness,Radius,Angle,Allowance
```

Allowance は曲げ代です。1 ファイルは一つの材料・金型の組合せです。板厚は一致する行を使用し、内 R・角度は表の格子内で補間します。範囲外や格子の不足を推測しません。［解除］で K 係数へ戻ります。選択部品へ適用すると、展開輪郭を保って曲げ代と直線部の境界を更新します。

板金ヘルプ

板厚・内側曲げ半径・曲げ角度ごとに、実測した曲げ代（BA）を登録する表です。材料や金型による違いを、K 係数の代わりに展開計算へ反映します。

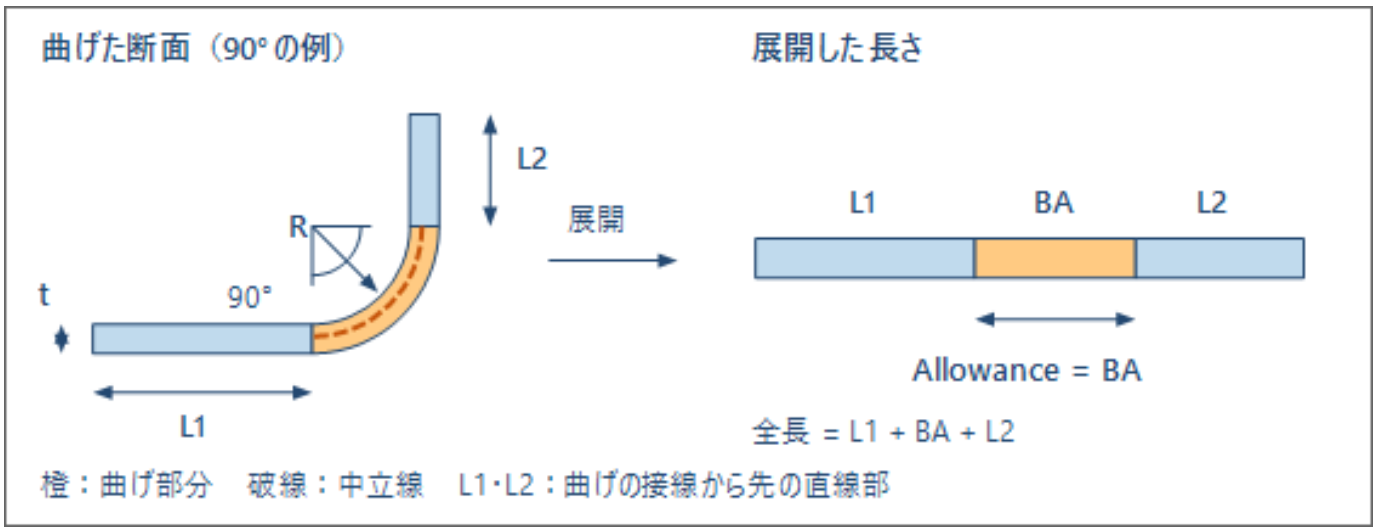


図 97　曲げた断面の橙色部分を展開すると曲げ代 BA になる。全長は直線部 L1 と BA と直線部 L2 の合計。

図の L1 と L2 は、曲げの丸みが始まる接線位置から先の直線部の長さです。外側の仮想交点から測る寸法ではありません。CSV の Allowance に入れるのは、橙色の曲げ部分だけを平らにした長さです。

1. CSV の書式

先頭行は次の 6 列を、この順序で記入します。見出しを日本語に置き換えたり、単位を数値に付けたりしないでください。

```
Material,Tool,Thickness,Radius,Angle,Allowance
DEMO,DEMO,1,1,45,1.099557
DEMO,DEMO,1,1,90,2.199115
DEMO,DEMO,1,1,135,3.298672
```

サンプル CSV を保存する

これは板厚 1mm・内 R1mm・K=0.4 で計算した説明用の数値です。材料・金型の実測値ではありません。製作には、お使いの加工条件で測定した値へ置き換えてください。

列	意味	例・単位
Material	材質の識別名	例：SPCC。同じファイル内では全行同じ名前
Tool	金型・加工条件の識別名	例：V8。同じファイル内では全行同じ名前
Thickness	板厚 t	mm、0 より大きい値
Radius	曲げの内側半径 R	mm、0 より大きい値
Angle	平らな状態から曲げる角度	度、0 より大きく 180 以下
Allowance	曲げ代 BA（曲げ部分の展開長さ）	mm、0 より大きい値

Excel では A 列から F 列へ入力し、CSV UTF-8（コンマ区切り）で保存します。小数点は「.」です。1 ファイルにつき材質と金型の組合せは 1 つです。材質・金型名から別の表を自動選択する機能ではなく、読み込んだ表が設定対象へ適用されます。

Angle は完成形の内側開き角ではありません。例えば内側の開きが 135° なら曲げ角度は 45° 、開きが 45° なら曲げ角度は 135° です。上曲げ・下曲げの表の角度は、どちらも正の値にします。

2. 外寸から引く「伸び値」をお持ちの場合

現場で「伸び」と呼んでいる値が、外寸の合計から引く量（曲げ控除 BD）なら、Allowance とは別の値です。1 か所の 90° 曲げでは次の関係になります。

例えば板厚 1mm・内 R1mm で、外寸合計から引く量が 1.800885mm なら、登録する BA は 4 − 1.800885 = 2.199115mm です。外寸 A=20mm、B=30mm なら展開全長は 48.199115mm です。A と B は外側の直線を延長した仮想交点まで測る寸法です。

90° 以外では、曲げ角度 θ に対して $BA = 2 \times (R + t) \times \tan(\theta / 2) - BD$ です（$0 < \theta < 180°$ 、θ は度）。180° の折返しにはこの換算式を使えません。内寸基準など別の定義の「伸び値」は、測り方を確認してから換算してください。

参考：K 係数から計算すると $BA = \theta \times \pi / 180 \times (R + K \times t)$ です。補正表を読み込んでいる条件では、表の BA が優先されます。

3. 読み込み手順と適用先

• ［板金］→［展開補正表］を開きます。

• ［設定対象］を選びます。［新規作成の既定値］は、この図面で今後作る板金用の設定です。既存部品は一括変更しません。［選択した板金］は、［CAD で選択］で指定した 1 部品へ適用します。

• ［補正表の操作］を［CSV を読み込む］にし、［適用］を押して CSV を選びます。［適用］を押すまでファイル選択画面は開きません。

• 完了メッセージを確認します。エラーなら材料・金型名、単位、適用する曲げの板厚・内 R・角度を確認します。

既存部品への適用では、現在の展開輪郭を保持して曲げ代と直線部の境界を変更します。完成形の外寸を固定して展開全長だけを変更する操作ではありません。

［補正表を解除］を選び［適用］すると、その設定対象の K 係数による計算へ戻ります。既存部品が部分展開中なら、先に再曲げしてください。ロール成形は K 係数を使用し、このプレス曲げ用の表は適用できません。

4. 行の追加と補間

• 板厚は一致するデータが必要です。板厚 1mm と 2mm を登録しても、1.5mm の板厚には補間しません。

• 内 R と角度は、登録範囲の内側だけで直線補間します。上の例なら R=1、角度 45°～135° に対応します。R=2 や角度 30° は範囲外です。

• 内 R と角度の両方の中間を求める場合、周囲 4 点が必要です。例えば R=1/2 と角度 60°/90° なら、その 4 通りすべてを登録します。

• 同じ板厚・内 R・角度の行を重複させないでください。範囲外や補間に必要な行の不足は、使用時にエラーになります。

• 現行の入力範囲は $\theta \times \pi/180 \times R \le BA \le \theta \times \pi/180 \times (R+t)$（K=0～1 相当）です。範囲外の実測値は、まず曲げ代と控除量、角度・半径の定義を確認してください。

この資料は JDesignCAD の CSV 読込み・補正表適用処理に合わせた説明です。

通常コマンドで板金を編集する

固定面を指定した部分展開

• ［板金展開］→［選択した曲げ］を選びます。

• 動かさない平面と、開く円筒状の曲げ面を選びます。曲げ面は複数選択できます。

• 展開された部分に通常コマンドで貫通穴・長穴などを加工します。開いた曲げ帯をまたぐ加工も可能です。

• ［板金再曲げ］で戻します。固定面の位置を維持します。

全展開では材料の境界エッジと曲げ中心線を選択基準にできます。同じ要素を再度クリックすると選択解除します。

展開図の曲げ番号・一覧表・更新確認

［展開図 作成・更新］と［展開データ出力］で、曲げ線・注記・曲げ一覧表を個別に表示できます。B 番号は曲げ管理と共通です。UP／DOWN は展開基準面からの曲げ方向、BA は曲げ代、WIDTH は曲げ軸の幅です。表は SM_BEND_TABLE に作成します。DXF・DWG の両方で出力します。

形状変更後に［板金チェック］を実行すると、既存の展開図が現在の部品と一致するかを確認します。変更されていれば［展開図 作成・更新］を実行してください。自動で図面を描き換える処理ではありません。

加工条件の確認

［板金チェック］で最小内 R と、穴から自由端・曲げまでの最小離隔を指定できます。0 はその項目を検査しません。曲げ帯内の穴も離隔不足になります。曲げや穴の有無に加え、形状の閉じ・板厚・展開の干渉を確認します。入力する限界値は材料・金型の基準を用いてください。加工時の割れ、肉薄化、スプリングバックや金型の進入経路の解析ではありません。

加工後の再編集・自動 45 度合わせ・寸法基準の使い方

箱全体の幅・奥行を［面の移動］で変更する

直角の箱形板金では、側板とその上のフランジを一緒に移動し、底板と隣の側板を連動して伸縮できます。上部の 45 度の継ぎ目、板厚、曲げ R、側板の高さ、フランジの奥行を保ちます。

• 曲げた状態で、通常の［面の移動］を開始します。

• 移動したい側の側板の広い面を選びます。表裏どちらでも使えます。その側板に付いた上部フランジの広い面を選んでも構いません。同じ側の側板とフランジを両方選ぶこともできます。

• 基点を指定し、底板に平行、側板に垂直な方向へ移動します。幅 100 mm を 120 mm にする例では、片側を外へ 20 mm 動かします。

• 確定すると、反対側を固定したまま底板・隣の側板・上部フランジが伸縮します。

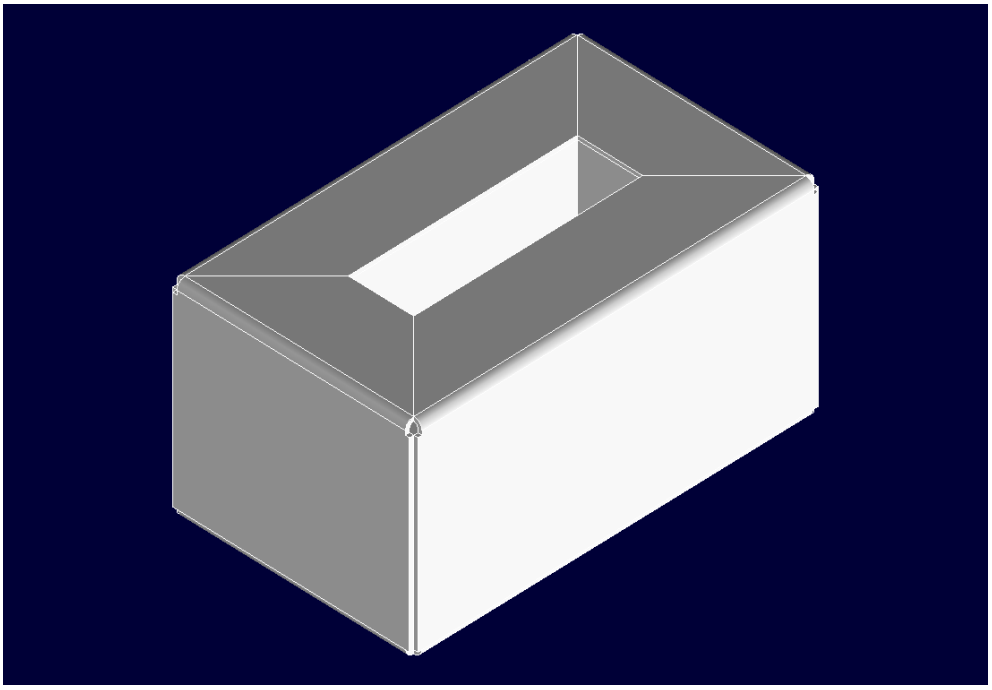


図 98　幅 100 ミリ、上部 4 辺に 45 度のフランジがある箱。

板金 19：変更前。幅 100 mm、奥行 60 mm。

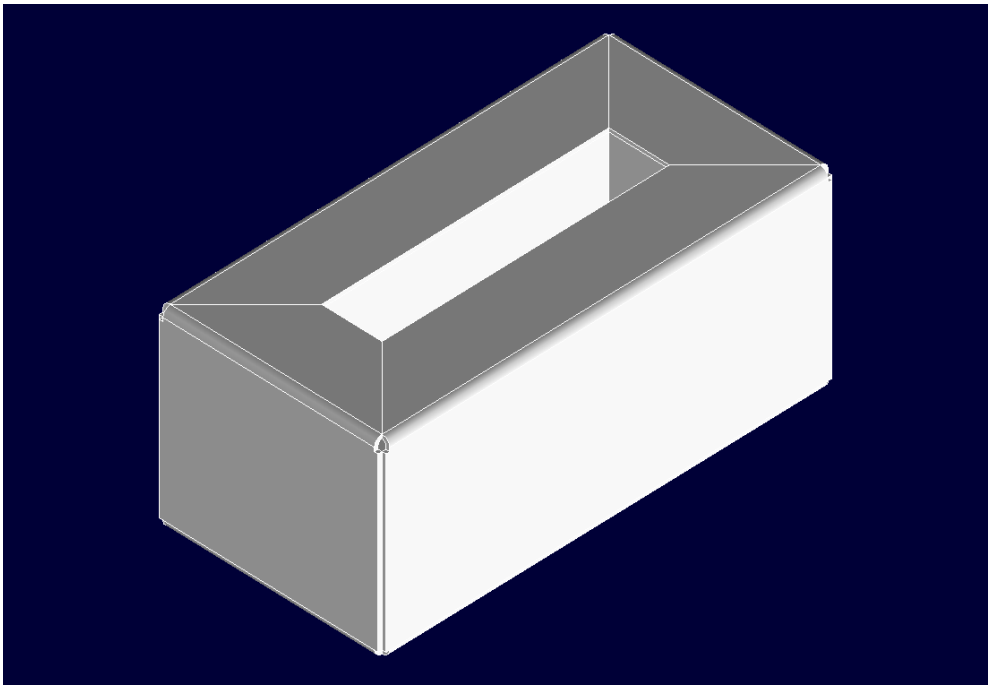


図 99　右側を 20 ミリ移動して幅 120 ミリへ広げた箱。上部の 45 度の継ぎ目を保持。

右側を 20 mm 移動：幅 120 mm。奥行と高さは変わりません。

移動中の水色のプレビューは、選択した面の移動先を示します。箱全体の再構成は確定時に行うため、プレビューで面が離れて見えても、確定後は接続を保って更新します。

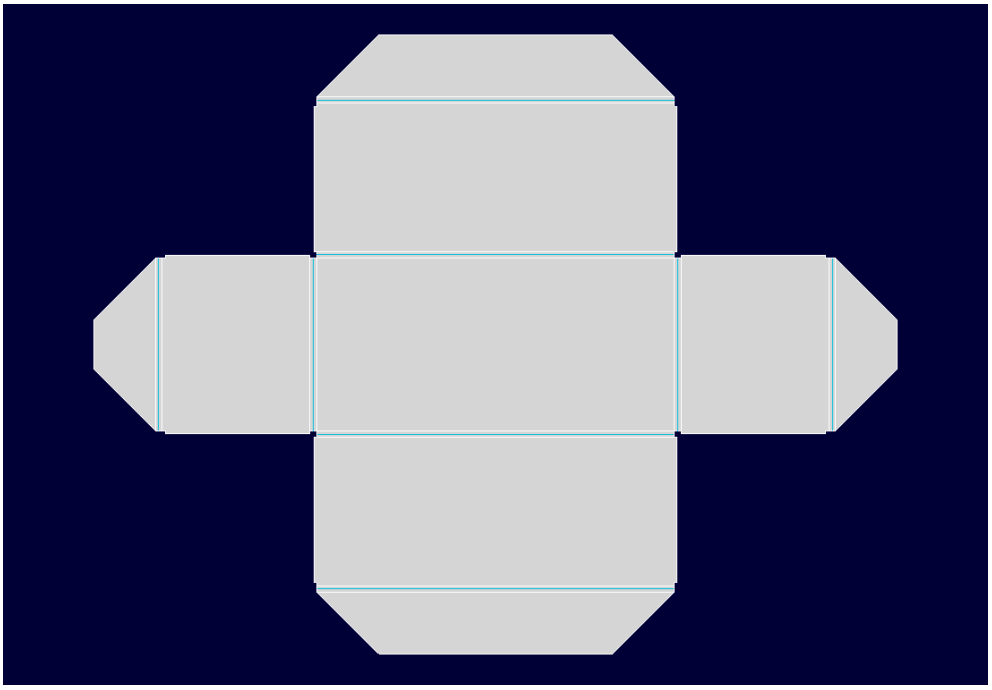


図 100　幅 120 ミリに変更した箱の展開形状。境界線と水色の曲げ中心線がある。

変更後の展開。再曲げ・保存・元に戻す／やり直しにも対応します。

この操作は、同じ側に属する面を選んだ直角の箱と、連続した直線の端部が対象です。幅を狭めることもできますが、材料が交差する場合や、端部の所属を一意に決められない複雑な切欠き・分割された曲げ帯は適用しません。通常の穴は大きさを保ち、移動する側板にある穴は側板と一緒に動きます。底板や隣の側板の穴はその位置に残ります。

下記の細い端面を選ぶ操作では、その端面を個別に編集します。箱全体の寸法を変える操作とは、選ぶ面が異なります。

展開した板の曲げ線を下書きの基準にする

［板金展開］を実行すると、曲げ帯の両端の境界線に加えて、曲げ帯の中央に水色の曲げ線を表示します。位置は、K 係数を含めて計算した曲げ帯の展開長の 1/2 です。表面と裏面の両方に作成します。

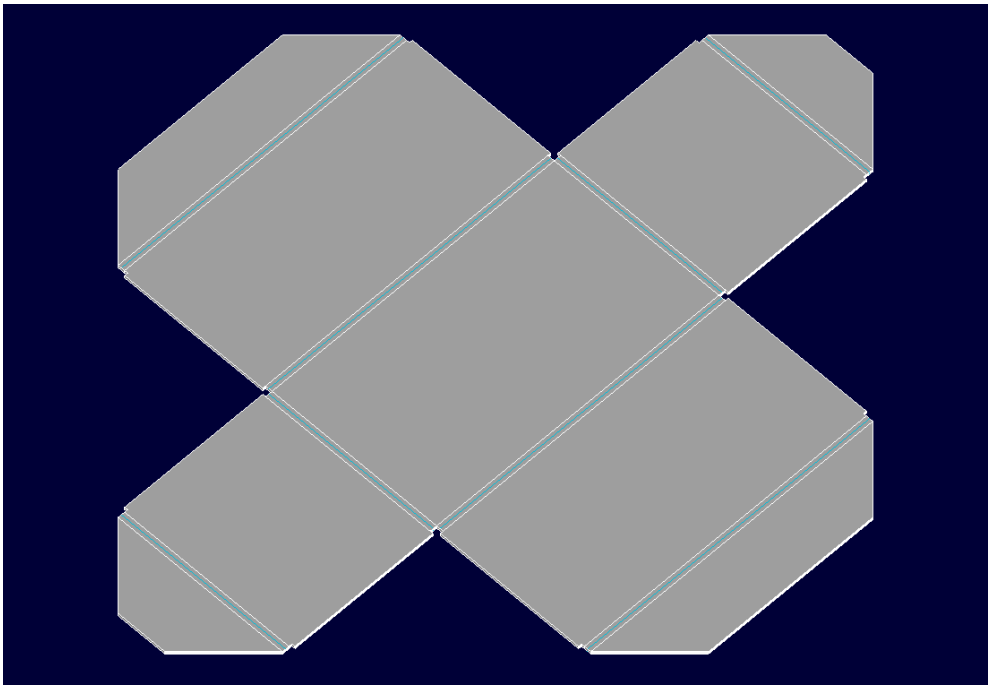


図 101　展開した箱。白い曲げ帯の境界線の間に水色の曲げ中央線がある。

白＝曲げ帯などの境界。水色＝曲げ帯の中央線。拡大すると 3 本を区別できます。

• ［板金展開］を実行します。旧版で展開済みの場合は、一度［板金再曲げ］してから展開し直してください。

• 板の表面上に下書きを置く場合は、作業平面をその表面に合わせます。

• 通常の下書き線コマンドで、水色の曲げ線を平行線などの基準に選びます。端点・中点・線上のスナップも使えます。

• 曲げ線を基準に輪郭や長穴を作図し、必要な貫通加工を行ってから［板金再曲げ］します。

中央線は実際に選べるエッジですが、板を切断する線ではありません。板厚・体積は変わらず、ソリッドは 1 個です。穴や切欠きにかかる部分は途切れます。ソリッドの移動・回転、保存・再読込、元に戻す／やり直しにも追従し、再曲げすると中央線の面境界はなくなります。

ご自身で作成した下書き線は通常の図形として残ります。［展開図作成・データ出力］では、この中央線が余分な切断線になることはありません。

板金登録されたソリッドでは、通常の［面の移動］［面のオフセット］［面のテーパー］を使って形状と展開情報を一緒に更新できます。以下の図は、板金 17 の複製に実際の処理を適用した結果です。

1. 底を固定して、側板と曲げの幅を変える

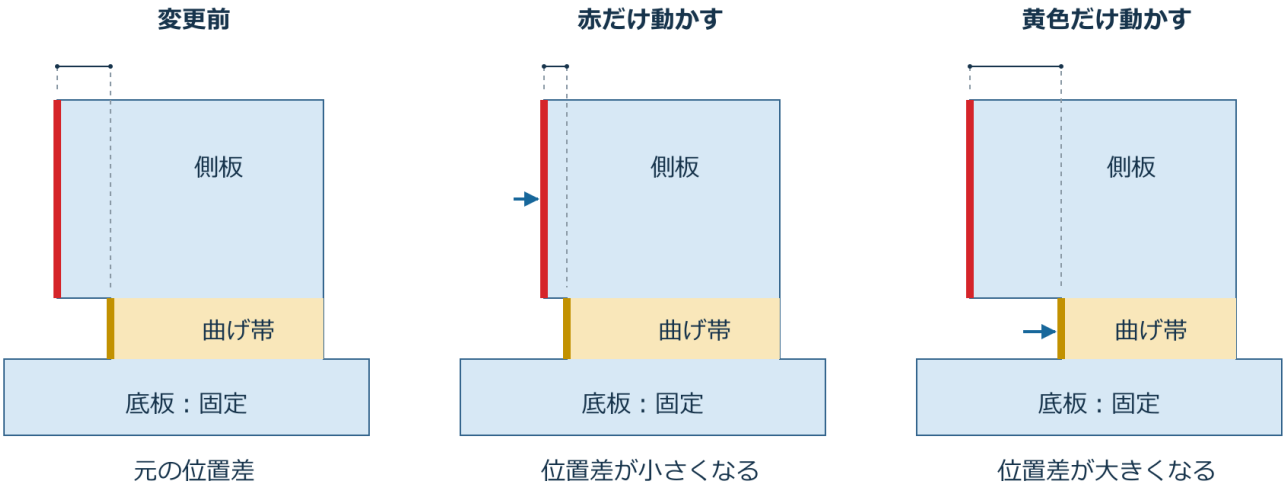
• 曲げた状態で［面の移動］または［面のオフセット］を開始します。

• 側板の細い縦端面（ご提示の赤い面）、または曲げ R の端の断面（黄色い面）を選びます。側板の広い面や曲げの円筒面は選びません。

• 面の移動では、曲げ軸と平行な幅方向へ移動します。面のオフセットでは、負の値で端を内側へ引き込み、正の値で外側へ伸ばします。

• 赤だけ選ぶと側板端だけ、黄色だけ選ぶと曲げ端だけが移動します。選ばなかった側と底板の平面部は固定します。端面間の位置差を変えて、根元の切り欠きを調整できます。

• 位置差を保って両方を移動したい場合は、赤と黄色を両方選択します。それぞれ一度ずつ同じ量で動きます。



展開した端部の模式図。矢印方向へ動かした例です。寸法・縮尺は実物と異なります。

図 102　赤い側板端と黄色い曲げ端を個別に動かし、端面間の位置差を調整する模式図。

赤と黄色の位置差は、側板全体の幅と分けて調整できます。面が一体の平面に結合されている形状では、その面に含まれる範囲が選択されます。

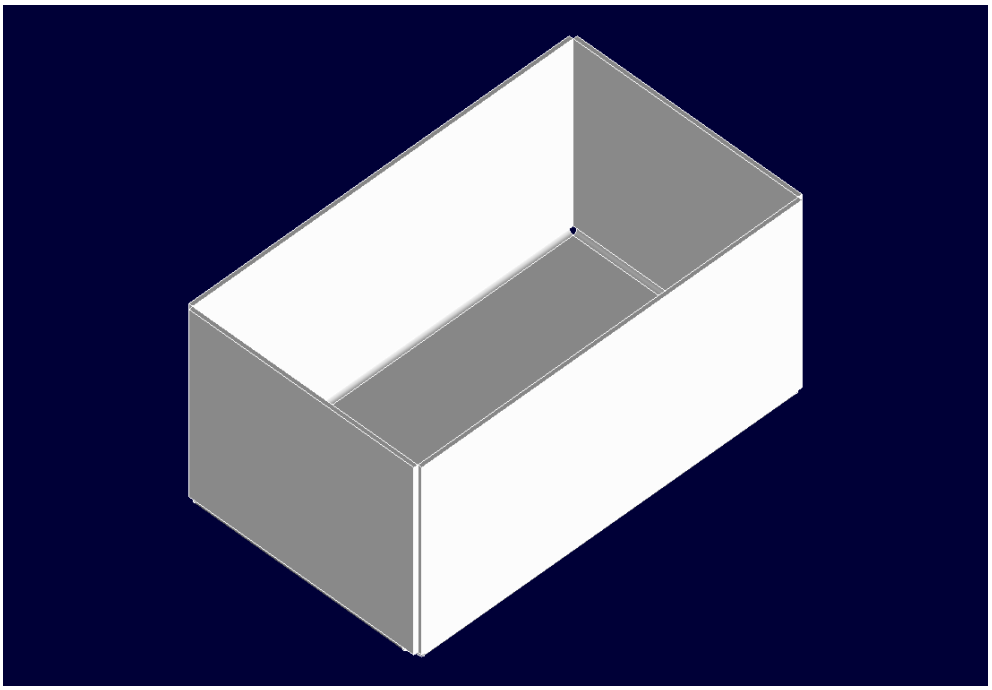


図 103　黄色の曲げ端だけを短くした結果。側板端と底板は固定。

曲げ端だけを内側へ動かした結果。根元の切り欠きが広がり、側板端の位置は変わりません。

他 CAD でも逃げの寸法は調整対象です。SOLIDWORKS は逃げ幅・深さ、Fusion は幅・深さ・残し量を指定できます。これは寸法調整機能の比較であり、各 CAD の通常の面移動がすべて同じ動作をするという意味ではありません。

伸ばすと隣の側板に干渉する場合や、短くしすぎて材料が分離する場合は適用しません。通常ソリッド全体の移動とは別に、端面を選ぶ操作です。

2. 展開したまま通常コマンドで編集する

• ［板金展開］で平らにします。

• 通常の面編集で、板厚の細い端面を選び、板の平面に沿って伸縮します。板厚を保った輪郭テーパーも使えます。

• 長穴は、閉じた長穴輪郭から板厚を貫く切除用ソリッドを作り、通常のブーリアン差で切り抜きます。曲げ帯をまたいでも構いません。

• ［板金再曲げ］を実行します。切除や輪郭変更を、底板・曲げ帯・側板へ引き継ぎます。



図 104　展開した十字形の板に、底板から曲げ帯をまたいで側板へ延びる丸端の長穴。

展開時：丸端の長穴を通常の切除で作成。

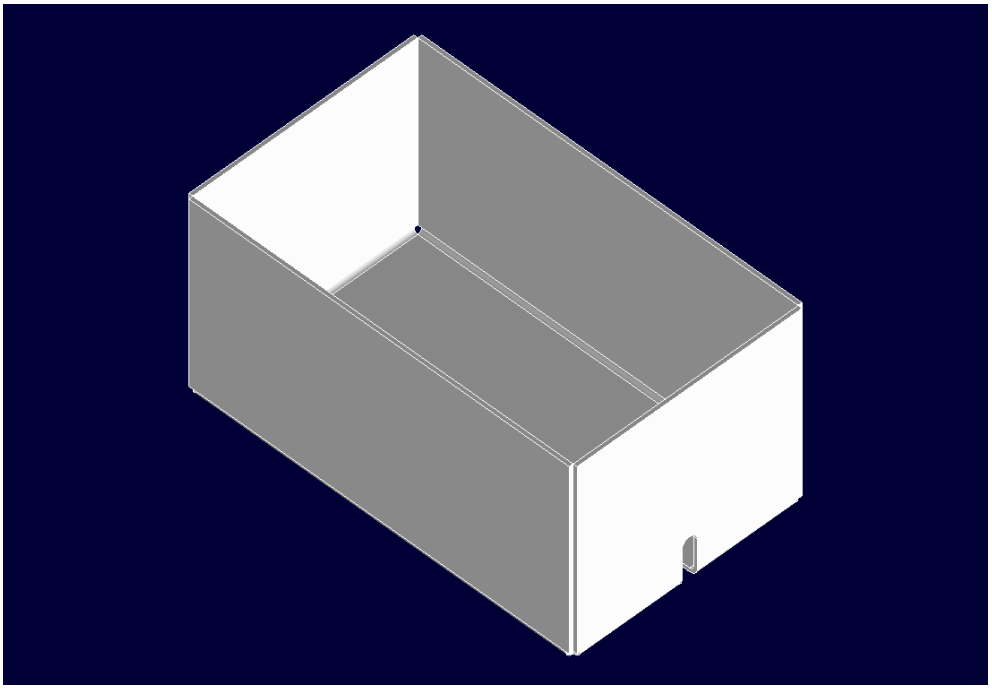


図 105　再曲げした箱の根元で、長穴が底板と曲げと側板を貫いて連続する。

再曲げ後：長穴が曲げに沿って連続します。

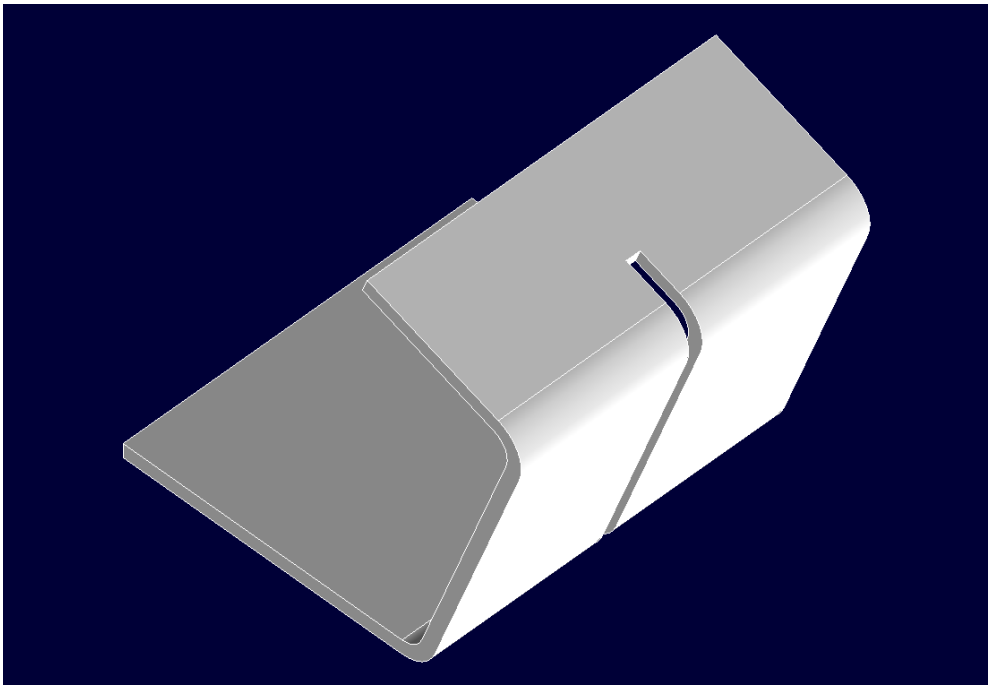


図 106　2 か所の曲げを連続してまたぐスリットの再曲げ結果。

複数の曲げをまたぐ切除も、展開時の輪郭から再構成します。

長穴を図面上に描くだけではソリッドに穴は開きません。貫通する切除を行ってください。伸ばした部分の所属が一意に決まり、一定板厚でつながる輪郭変更が対象です。独立した材料の追加や、複数の曲げ領域に重なる追加は適用しません。

3. フランジを台形にする

• 通常の［面のテーパー］で、フランジの細い側端面を選びます。両端を同時に選ぶこともできます。

• 曲げた状態では、フランジの平面と曲げ R の境界にある長い直線を基準に選べます。展開した状態では、曲げ帯との境界線や、端面の根元にある板厚を横切る短い辺も使えます。

• ＋45 度で先端側を狭め、－45 度で広げます。板厚と根元の位置を保ち、選んだフランジの輪郭を変更します。

図解：箱の上部フランジを曲げたまま 45 度に狭める・広げる／展開時の境界エッジ

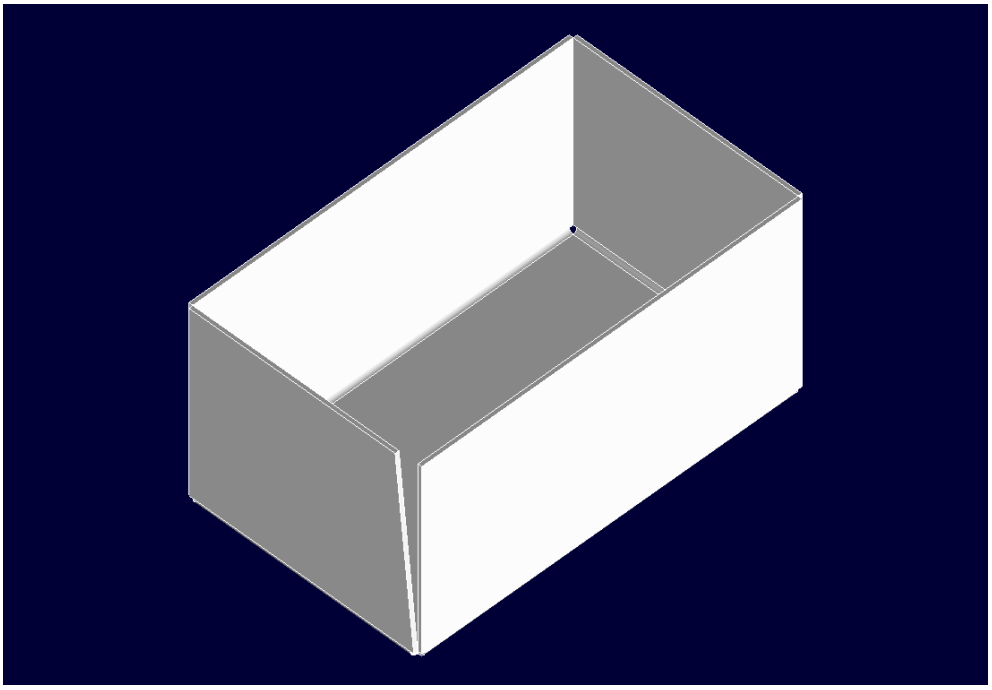


図 107　箱の一側板の端が根元から上端へ斜めになり、板厚を保った台形になる。

曲げ状態：側端面を傾けた結果。

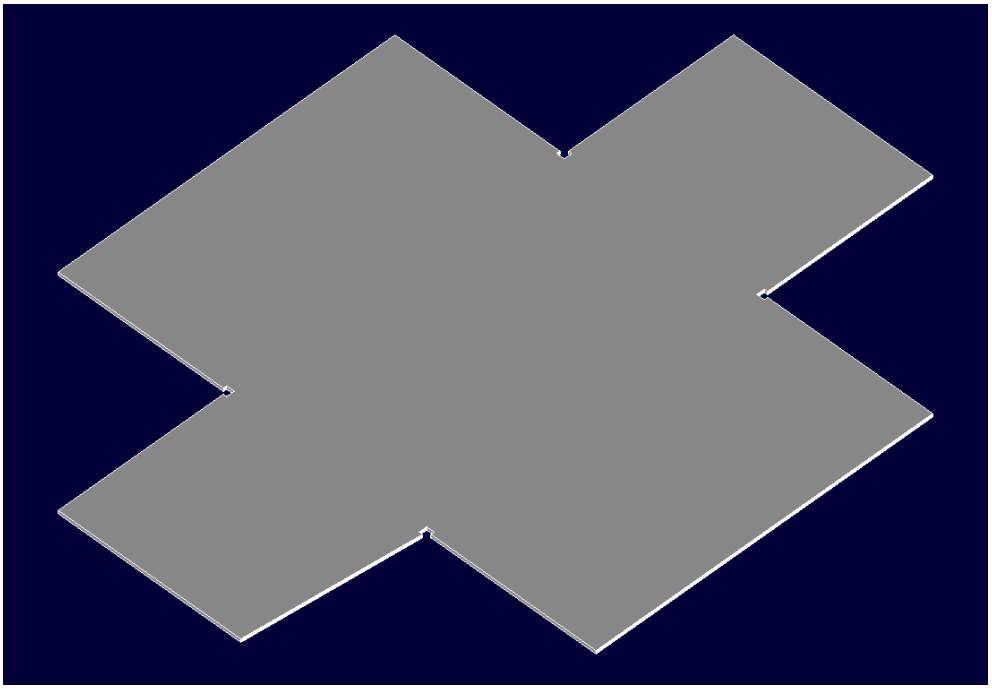


図 108　台形の側板輪郭を保持した展開結果。

展開状態：台形の輪郭も保持します。

ここで対応するのは輪郭を台形にするテーパーです。広い表面・裏面を傾け、板厚が先細りになるテーパーは再曲げの対象外です。

4. 曲げたまま、押し出しで角の段差を埋める

• 角の段差にある細い端面を選び、通常の押し出しで材料を追加・結合します。側板の表裏の位置を変えず、板厚を一定に保ちます。

• ［板金展開］を実行します。平面部の追加・切除を読み取り、展開輪郭へ反映します。

• ［板金再曲げ］で、追加・切除した形状を保ったまま戻します。押し出し後に保存した JDC からも展開できます。

板金 24 の角の段差を 0.5 mm／1 mm 埋める操作で確認しています。平面部だけを貫く切除にも対応します。板厚が変わる盛り上げ・止まり穴、曲げ R 自体を変更する加工、別の板へつながって所属が曖昧になる追加は取り込めません。曲げ帯をまたぐ長穴は、展開して加工してください。

保存・元に戻す・対応範囲

通常の面編集では形状と板金情報をまとめて更新します。［元に戻す］［やり直し］と JDC 保存・再読込に対応します。選択・数値入力中は重い再構成を行わず、確定時に接続・干渉を検査します。

• 止まり穴、板厚変更、材料の分離、曲げ後の干渉は対象外です。

• 曲げ帯の加工後も角度・R の変更とフランジ追加ができます。展開輪郭と穴位置を保持するため、加工済みの曲げの直線部長さは自動計算します。逃げの再生成と自動コーナー調整は加工前に行ってください。

• 通常の押し出し・結合・切除による平面部の変更は、次の［板金展開］時に取り込みます。曲げ形状と板厚が保たれ、再構成した形状が編集後の全体形状と一致する場合が対象です。

固定面を選ぶ部分展開・追加の通常編集

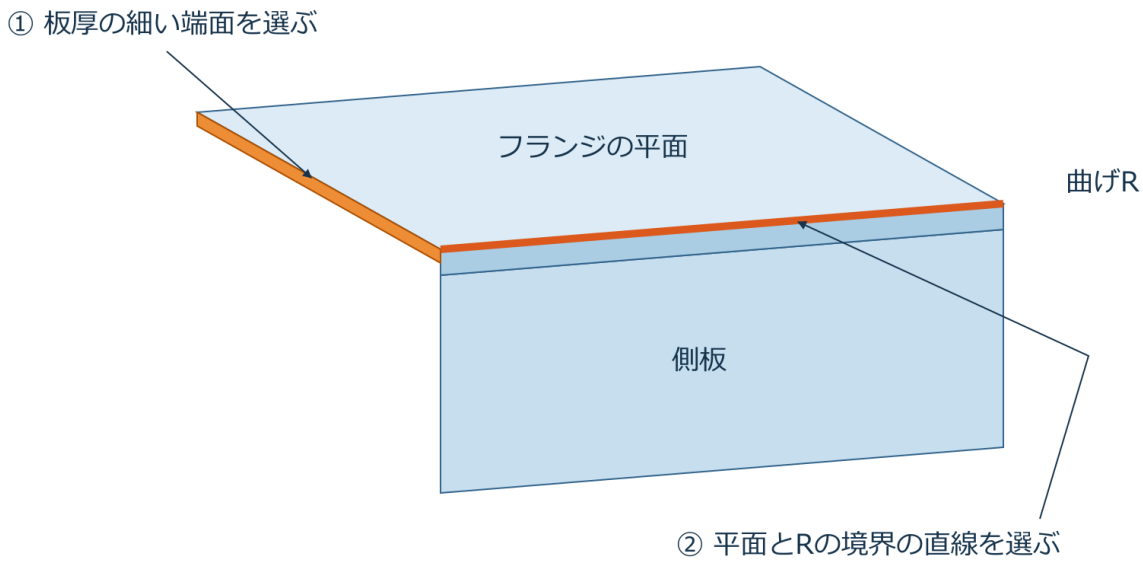
フランジの端を 45 度に狭める・広げる

箱の上部 4 辺を作りたい場合は、エッジフランジの［端部の形状］で両端 45 度を指定して作成してください。既存のフランジと重なるため追加できない場合の、範囲設定を含めた手順です。

通常の［面のテーパー］で、曲げた状態のフランジ端を狭めることも、外へ広げることもできます。上部に折り返したフランジや、外向きに折ったフランジが対象です。板金 18 を複製して検証した例を示します。

曲げたまま操作する

- ［面のテーパー］を開始し、フランジの板厚の細い端面を選びます。両端を同じ角度にする場合は両方を選びます。
- ［基準境界エッジ］で、フランジの平面と曲げ R の境界にある長い直線を選びます。R の端にある円弧や、フランジ先端の自由な辺ではありません。
- 角度を入力して確定します。＋45 度で先端側を狭め、－45 度で先端側を広げます。根元の位置と板厚を保ちます。



選択箇所の模式図。広い平面をテーパー面には選びません。

図 109　細い端面を選び、フランジの平面と R の境界線を基準にする。

①テーパーする端面 →②根元の直線 → 角度、の順番です。側面が見える方向へ回転・拡大して選択してください。

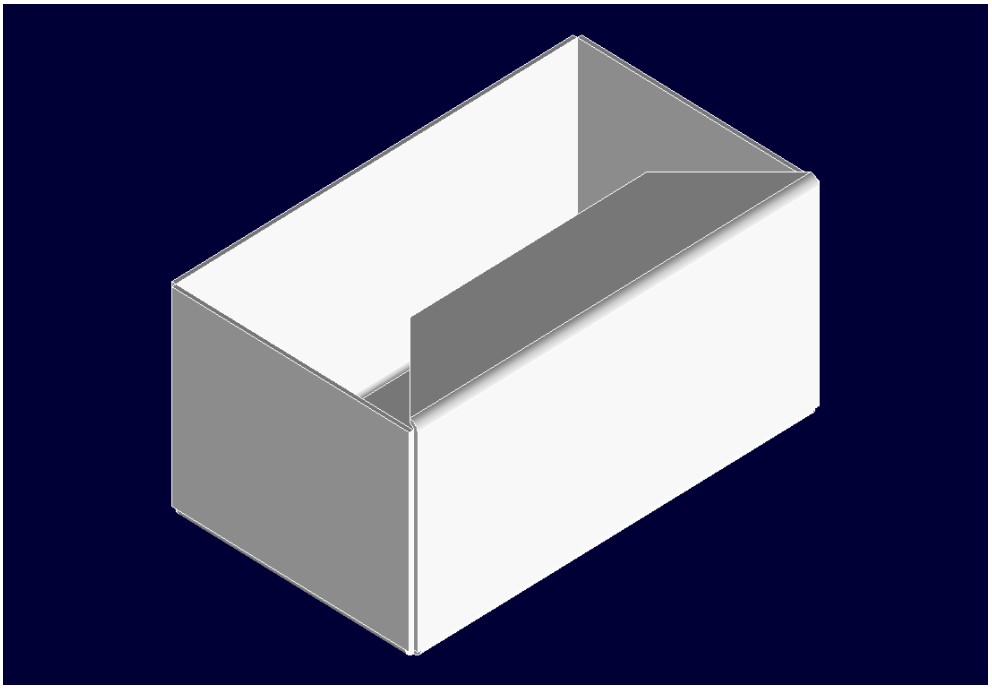


図 110　両端を 45 度に狭めた箱の上部フランジ。

＋45 度：根元を残し、両端の先端側を狭めます。

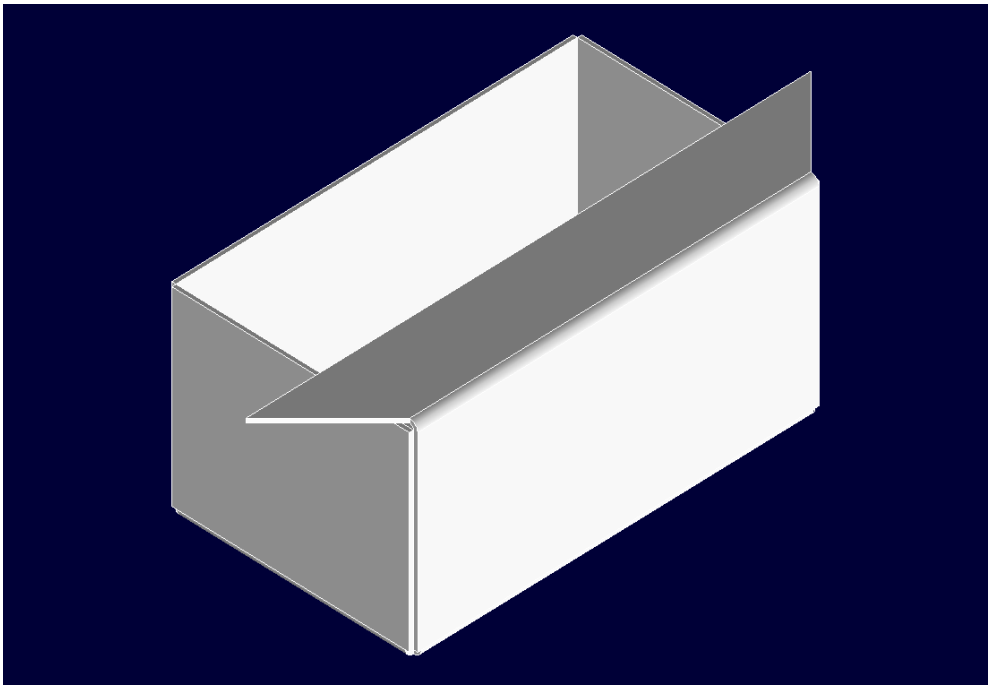


図 111　両端を 45 度に広げた箱の上部フランジ。

－45 度：材料を増やし、両端の先端側を広げます。

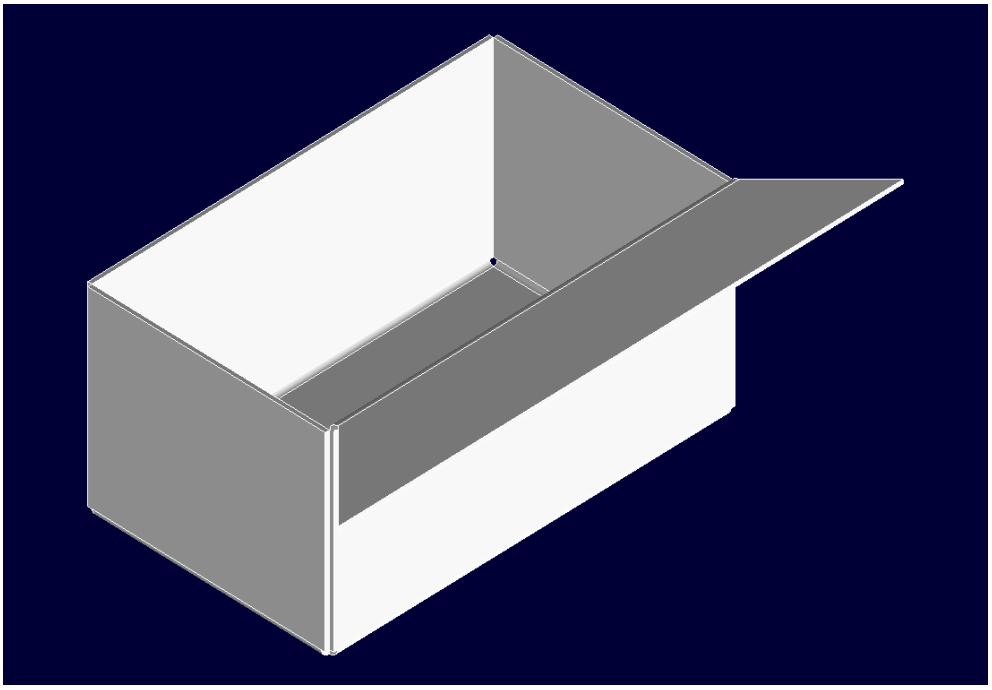


図 112　外向きの折り返しフランジの両端を 45 度に広げた結果。

外向きに折ったフランジでも、同じ操作で広げられます。正負は画面の左右や上下ではなく、根元から離れる側の狭まり・広がりです。

展開して編集する場合

• ［板金展開］を実行します。底板・側板・曲げ帯・上部フランジの境界を、実際に選択できる面とエッジとして残します。

• 通常の［面のテーパー］で、対象フランジの細い端面を選びます。基準は、曲げ帯とフランジの境界の直線、または端面の根元にある板厚を横切る短い辺です。

• ＋45 度／－45 度を指定して確定します。［面の移動］［面のオフセット］でも、選択した部分の幅を調整できます。

• ［板金再曲げ］を実行します。広げて追加された材料も、狭めて除かれた材料も反映します。

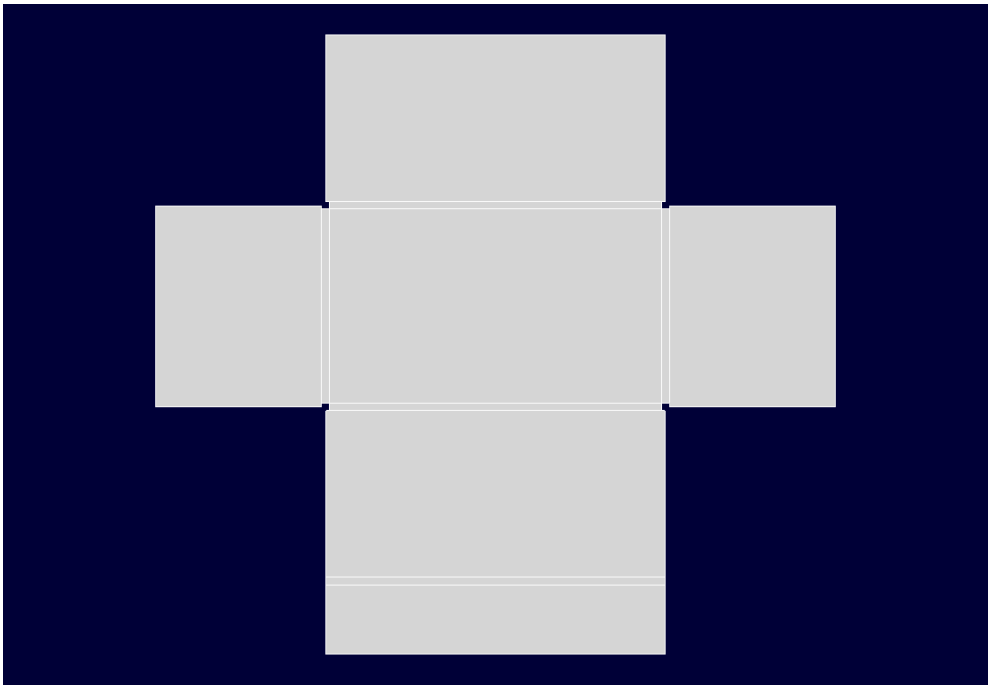


図 113　展開した箱。底板と各側板の間の曲げ帯、下側の側板と上部フランジの間の曲げ帯が実線で分かれている。

2 本の境界線の間が曲げ帯です。境界線は切断線ではなく、ソリッドは 1 個のままです。

古い版で展開して保存した形状に境界がない場合は、一度［板金再曲げ］してから［板金展開］してください。加工用の［展開図作成］では、外周・穴を切断線として出力し、この編集用の面境界を余分な切断線にはしません。

操作の範囲

• この操作はフランジの平面部分の輪郭を変えます。基準にした根元と曲げ R は維持します。曲げ R の端まで斜めに加工する場合は、展開して曲げ帯も含む貫通加工を行います。

• 隣り合うフランジの自動留め加工や、隙間寸法の自動調整ではありません。それぞれの端面を編集し、接触・干渉と必要な隙間を確認してください。上部 4 辺を一度に作って自動で 45 度に合わせる機能はまだありません。

• 板厚を変えるテーパー、接続が切れる変更、材料同士が重なる変更は適用しません。曲げたままの任意のブーリアン演算を追跡する機能ではありません。

• 保存・再読込と［元に戻す］［やり直し］に対応します。曲げ帯に加工を入れた場合は、角度・R の後からの変更などに制限があります。

板金の加工後編集・コーナー自動調整・寸法基準

板厚、内側曲げ半径、K 係数は図面または現在の板金設定に合わせます。曲げ端と隣接フランジの接続では必要な逃げを残し、不要な突起や重複する逃げを避けます。逃げを消して材料を重ねる操作は、製作可能な接続の代わりにはなりません。

展開・再曲げ後は、穴と切欠き、曲げ端、隣接面の干渉を確認します。通常編集で追加した板金加工は対応する板厚方向と曲げ位置でのみ保持できます。板厚変更、盲穴、対応していない斜めの曲げ部加工などは拒否される場合があります。失敗時は元形状を維持し、エラーを確認して加工方法を変更してください。

MCP の sheet.flange は startDistance と endDistance でエッジの一部区間を指定できます。展開図をレイアウトへ配置する場合は layout.create を使い、寸法と文字付き引き出し線を入れて view.capture で確認します。

通常コマンドで加工した板金の再編集と、箱上部のフランジ作成を補助する機能です。いずれも［適用］時に計算し、成功した場合だけ図形を置き換えます。元に戻す／やり直しに対応しています。

長穴や切欠きを残したまま曲げを編集

- ［板金展開］して、通常の切除で貫通穴・長穴・切欠きを作成します。曲げ帯をまたいでも構いません。
- ［板金再曲げ］で折り曲げた状態に戻します。
- ［曲げ・フランジ編集］で円筒状の曲げ面を選び、角度や内側半径 R を変更します。

展開輪郭と穴の位置を保持して、曲げ帯とその先の平面部の境界を計算し直します。加工済みの板や線で作った曲げでは、直線部の長さ L は自動計算です。角度・R が大きくなれば直線部が短くなります。加工済みの曲げを残したまま、自由な直線エッジに［エッジフランジ］を追加することもできます。

板厚変更、材料の分離、別の曲げの付け根を越える変更、曲げ後の干渉は適用できません。加工済みの曲げに対する逃げの再生成や、自動コーナーによる再作図は対象外です。穴や輪郭の通常編集は引き続き使えます。

箱の上部 4 辺を自動で 45 度に合わせる

- ［エッジフランジ］で側板の上端の直線エッジを選びます。
- 範囲は［エッジ全体］または［開始・終了位置を指定］、端部は［直角］、［コーナー自動調整］は［自動で合わせる］を選びます。
- 長さ、角度 90 度、内側 R、隙間（例：0.2 mm）を指定して［適用］します。
- 隣接する上端で同じ操作を繰り返します。隣のフランジがある角を検出し、両方の端を調整します。最初の 1 辺には隣接相手がないため、直角の端が残ります。

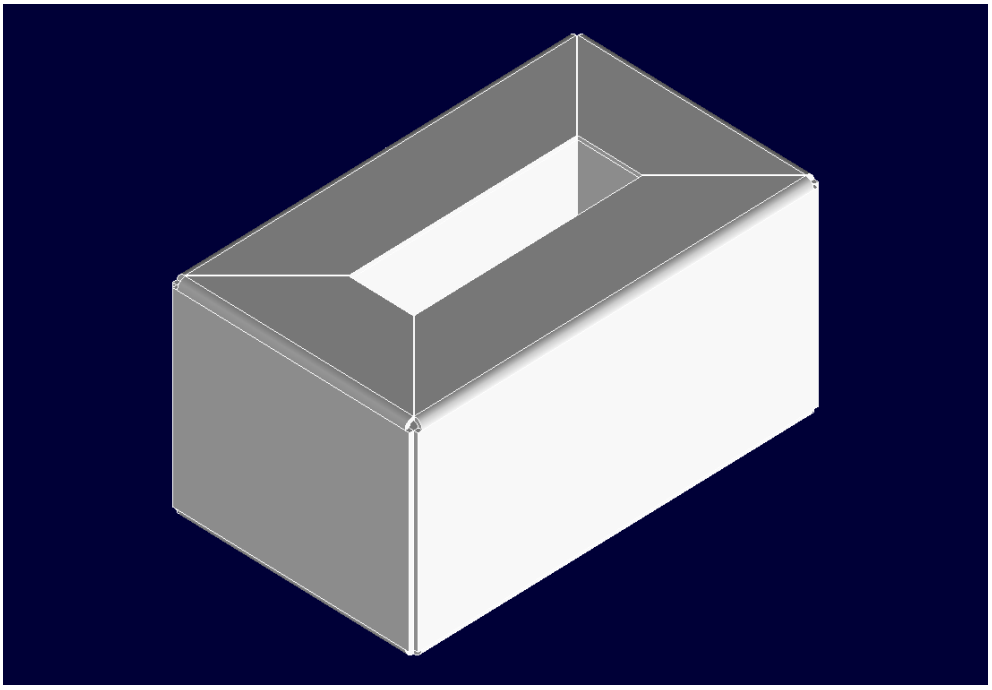


図 114　自動合わせした内向きの箱上部フランジ

内向き：先端を切り詰めて合わせます。

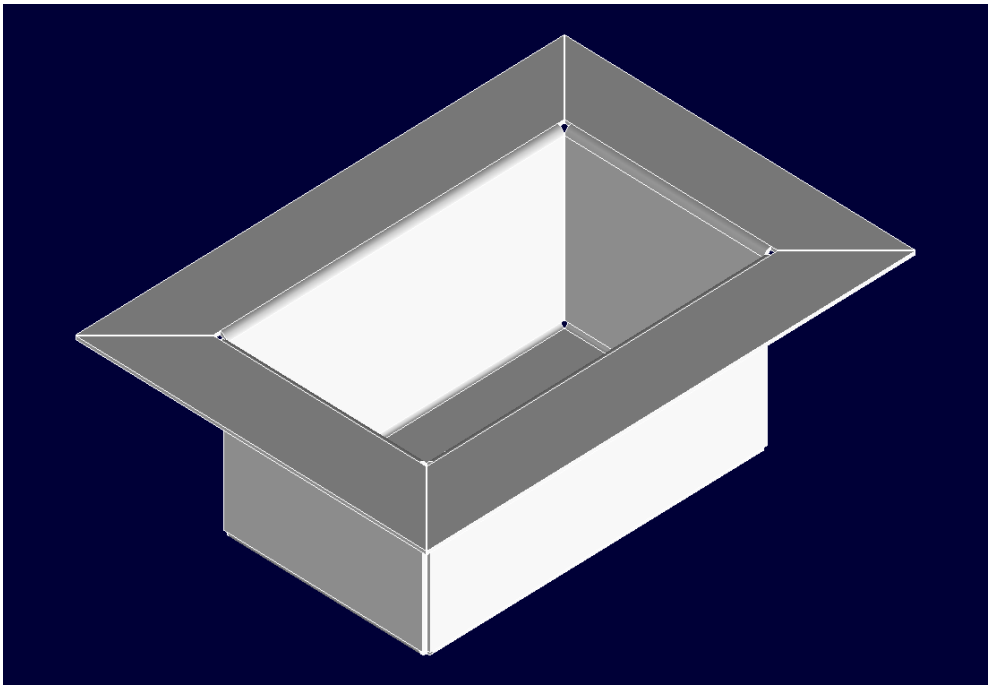


図 115　自動合わせした外向きの箱上部フランジ

外向き：必要な部分を広げて合わせます。外側の上端エッジを選びます。

45 度の合わせ位置は折り曲げた形から求めます。隙間は合わせ線に直角に測った距離です。R 部分は必要な端の退避を行います。曲げ R 部分の開口幅が、平面部の隙間と同じになるわけではありません。

左右に余白を残す場合

エッジ長が 57.9 mm で両端を 1 mm ずつ短くする場合、［開始・終了位置を指定］で開始 1 mm・終了 56.9 mm にします。［端部の形状］は［直角（長方形）］のまま、［コーナー自動調整］を［自動で合わせる］にしてください。範囲指定で親板側に作られる長方形・丸形の逃げを残したまま、隣接フランジと 45 度に合わせます。

両端の余白と、斜めの合わせ目の隙間は別の寸法です。合わせ目を 1 mm にする場合は［隙間幅］にも 1.0000 mm を入力してください。開始・終了位置だけを 1 mm ずつ短くしても、［隙間幅］が 0.1000 mm なら、合わせ目の間隔は 0.1 mm です。両端に 1 mm の余白を残し、隙間を 0.1 mm または 1 mm にする条件は、内向き・外向きとも作成・展開・再曲げを確認しています。長さ 19.1 mm での作成と、既存の全幅フランジの横への追加にも対応しています。選択範囲は曲げ帯の最大幅を指定します。合わせ目に応じて曲げ端はさらに退避し、外向きの平面部は必要な分だけ張り出します。

作成後に隙間を変更

［コーナー処理］→［45 度合わせ（同一平面）］で、隣接フランジの広い平面を 2 枚以上選び、隙間を指定します。箱上部の 4 面を選べば 4 角をまとめて調整できます。隙間を広げる／狭める両方に対応します。

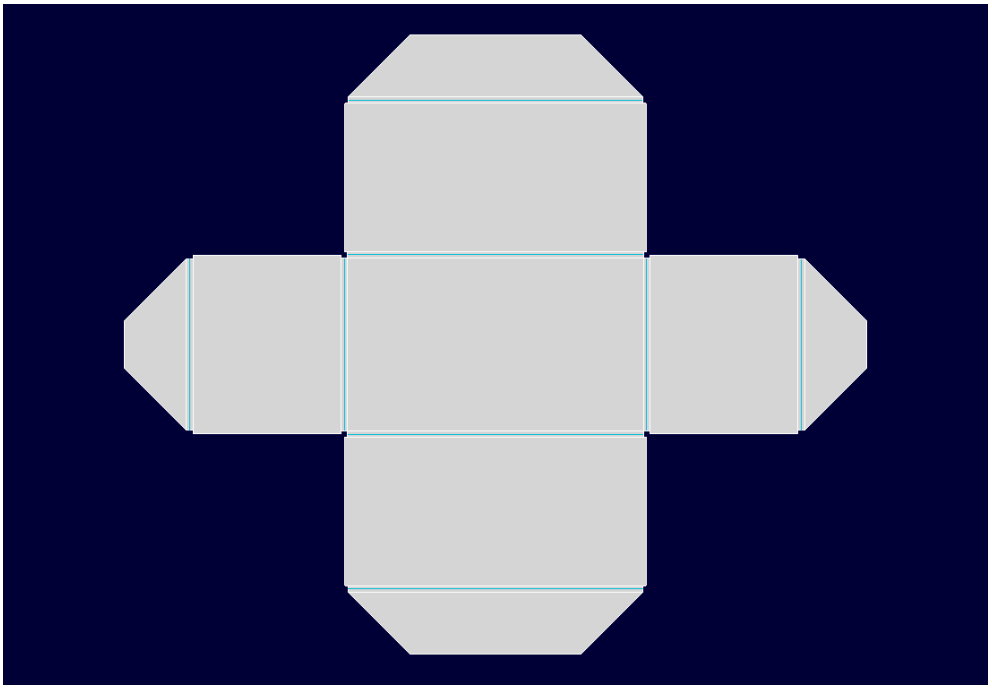


図 116　自動合わせした箱の展開形状と曲げ線

展開形状にも合わせ部分を反映します。境界エッジと曲げ中心線を保持します。

同じ平面にある、直交して隣り合うフランジが対象です。範囲指定で親板側に作られる端部の逃げは併用できます。通常コマンドで個別に輪郭加工した形状や、曲げ帯・フランジの幅を削る旧形式の逃げは自動調整の対象外です。非直交の任意角、段違い、重ね継ぎの自動作成は対象外です。［突合せ］は従来の曲げ端を退避させる処理です。

曲げ位置と長さの基準

従来の形状と同じにする場合は、位置＝曲げ開始（接線）、長さ＝曲げ先の直線部にします。旧図面の寸法の解釈は変更しません。位置・長さの基準欄を選ぶと、ダイアログの断面図が切り替わります。

曲げ位置	選択したエッジ／線との関係
曲げ開始（接線）	曲げ帯の開始を合わせます。
内側の仮想交点	曲げ前後の内側の直線を延長した交点を合わせます。曲げ開始は R × tan(角度/2) だけ手前です。

曲げ位置	選択したエッジ／線との関係
外側の仮想交点	外側の直線を延長した交点を合わせます。曲げ開始は（R＋板厚）× tan(角度/2）だけ手前です。
展開時の曲げ中心	展開した曲げ帯の中央を合わせます。開始は曲げ帯の展開長の半分だけ手前です。

エッジフランジは手前側の板を必要な幅だけ曲げに使います。その部分に板がない場合や、穴・別の曲げがある場合は適用しません。［線で曲げる］にも同じ位置基準が使えます。

長さの基準	直線部 L の計算
曲げ先の直線部	入力値そのまま。
内側の仮想交点	入力値 − R × tan(角度/2)。
外側の仮想交点	入力値 − (R＋板厚) × tan(角度/2)。

例えば板厚 2 mm、内側 R3 mm、90 度で外側寸法 20 mm なら、直線部は 15 mm です。作成時に選んだ長さの基準は保存され、通常のフランジ編集でも同じ基準で表示・入力できます。加工済みの板は直線部の自動計算表示になります。曲げ位置の基準は作成時の配置指定です。作成後の角度・R 編集では曲げ開始位置を固定します。90 度以外では、寸法はフランジに沿った距離です。画面の上下方向や垂直な高さではありません。計算した直線部が 0 以下になる寸法は適用しません。

ロール・ロフト：丸丸・角丸・角角継手

ロール・ロフト

同軸円筒・円すいのロール、偏心にも対応する丸丸・角 R 付き角丸／角角の滑らかな成形と 2 スケッチからの作成、丸丸・角丸・角角の三角法ロフトを作成できます。偏心量・継ぎ目・分割数を指定し、展開・再曲げ・寸法編集に対応します。三角法は中立面寸法を使用し、有限の曲げ R・伸び補正は含みません。操作手順・寸法基準・制限事項をご確認ください。

［板金］→［ロール・ロフト］で作成します。STD／PRO で使用できます。丸→丸・角 R 付き角→丸・角→角は［成形方法］で滑らかな成形と三角法（分割面）を選べます。滑らかな成形はロール・型成形向け、三角法は各三角形の辺長を保つ近似展開です。三角法には有限の曲げ R やプレス曲げ補正を含みません。

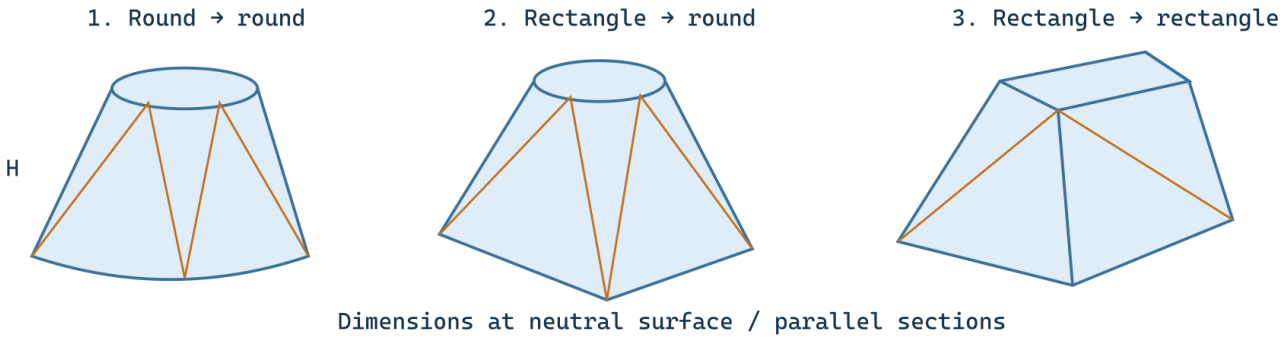


図 117　Three loft section families and neutral dimensions

切断位置を自動・指定で選ぶ

［ロール・ロフト］の［切断位置］を設定します。丸―丸・角―丸・角―角の滑らかな成形／三角法と、円筒・円すいのロールに共通です。［自動］は従来の位置を使います。既存図面も、変更しなければ従来の位置を維持します。

- 断面寸法を入力するか、2 つの断面スケッチを選択します。
- ［切断位置：指定］に切り替えます。
- ［切断方向（度）］を入力します。下側断面の中心から見て、局所＋X が 0°、＋Y が 90°、－X が 180°、－Y が 270° です。角 R の中心を基準にした角度ではありません。
- 画面上で決める場合は［切断方向を CAD で指示］のボタンを押し、下側断面の輪郭上の点を指示します。角度が自動入力されます。中心点は方向が定まらないため選べません。
- ［適用］して作成し、［板金展開］で確認します。［既存ロール板を編集］でも、同じ項目で切断位置を変更できます。

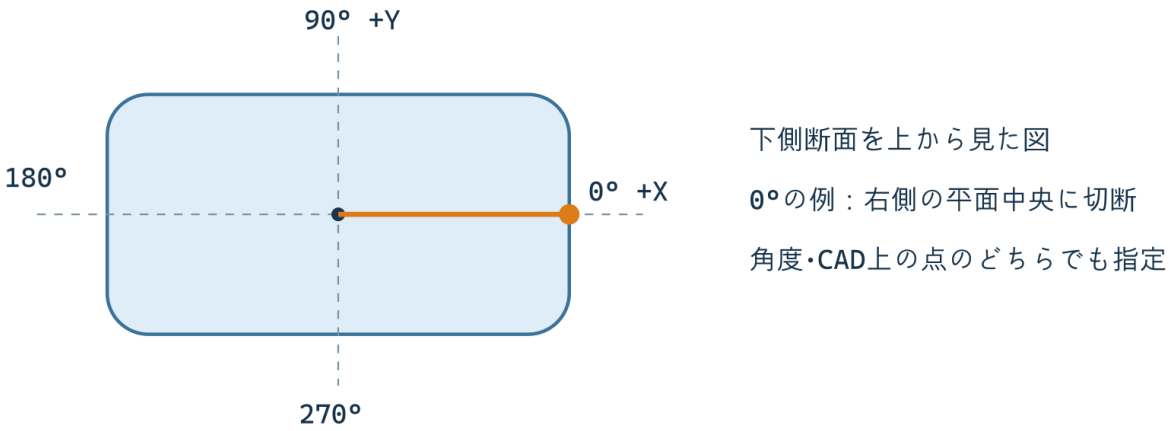


図 118　下側断面を上から見た切断方向。右 0 度、上 90 度、左 180 度、下 270 度。

寸法入力時の局所軸は、作成時のワールド X／Y です。スケッチから作る場合は下側断面の軸を使い、移動・回転後の編集でもその軸を維持します。画面の左右と一致しない場合は CAD 上の点で指定してください。指定した下側の位置から、対応する上側断面までの母線に沿って開きます。部品全体を回転する操作ではありません。

指定モードの形状	隙間の基準
角一丸・角一角、三角法ロフト	展開上で切断線に直角な幅。切断線の両側から半分ずつ除きます。角一丸の平面中央でも上端まで開きます。
滑らかな丸一丸	小さい円側の中立面端点間距離。もう一方は半径比で広がります。
円筒・円すいのロール	巻き角度で開き幅を指定します。切断方向は、開いている角度の中央です。

展開後の穴・切り欠きがある場合は、展開板の基準位置からの距離と加工寸法を保ちます。切断位置を大きく変えると展開輪郭も変わるため、加工位置が新しい材料に収まらないと変更を適用できません。入力中は形状計算せず、［適用］時に確認します。

滑らかな丸―丸を作る

- ［断面の種類：丸→丸（偏心可）］を選びます。
- ［成形方法：滑らかな成形（ロール・型成形）］を選びます。
- 上下の中立面半径、高さ、上側中心のずれ X／Y、板厚、K 係数を指定します。同心は X＝Y＝0 です。
- ［継ぎ目の隙間］は小さい円側の中立面上の端点間距離です。継ぎ目は母線に沿うため、大きい円側では半径の比に比例して隙間が広がります。
- ［適用］後、［板金展開］で切断輪郭を確認します。分割数の指定は不要です。

立体は連続した曲面で、展開には三角形の分割線を付けません。同心の円すいは正確な円弧 2 本と直線 2 本、同心の円筒は長方形です。偏心した円すい・円筒は長さを保って開き、円弧とは限らない輪郭をスプラインで出力します。偏心の展開曲線の計算許容差は 0.00001 mm です（加工機・材料の成形精度を表す値ではありません）。DXF／DWG も円弧・スプラインを維持し、SM_CUT レイヤーへ出します。

半径・高さは中立面基準です。実際の内外面は中立面から板厚方向へ－K×t／＋(1－K)×t だけ移した面です。内寸・外寸基準ではなく、偏心の場合の板厚端面は XY 平面上の円とは一致しません。内径を直接指定する同心ロールには、従来の［円筒・円すい（滑らかなロール)］を使えます。

既存の丸→丸も［既存ロール板を編集］で選び、成形方法を変更できます。展開後の穴・切り欠きを保持する寸法変更は、下記の「加工後に寸法・角 R を変更する」をご覧ください。切り替える際は継ぎ目の隙間の基準が異なる点を確認してください。角→丸の手順は下記をご覧ください。角→角の滑らかな成形は、下記の専用手順をご覧ください。

角 R 付きの角―丸を滑らかに作る

- ［断面の指定方法：寸法を入力]、[断面の種類：角→丸（角 R・偏心対応)］を選びます。
- ［成形方法：滑らかな成形（ロール・型成形)］を選びます。
- 矩形の幅・奥行き・角 R、円の半径、高さ、偏心 X／Y を指定します。角 R は 0 より大きく、幅・奥行きの半分未満にします。
- 板厚、K 係数、継ぎ目の隙間を指定して［適用］します。[板金展開］と［板金再曲げ］で切り替えます。

四隅を円弧にし、平らな側面と滑らかな角の曲面を接線でつなぎます。上下の中心がずれた偏心形状にも対応します。丸側が大きい形、角 R と丸側の半径が同じ形にも対応します。展開は長さを保った曲線輪郭になり、三角形の分割線は付きません。DXF／DWG もスプラインと直線を保持します。

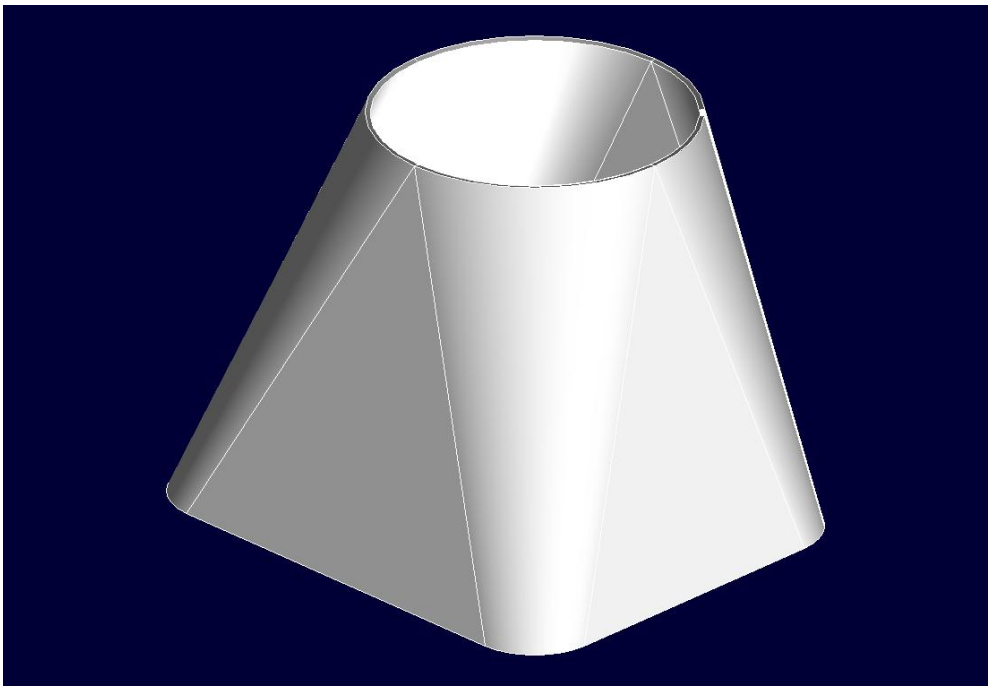


図 119　角 R 付き偏心角丸の実際の作成結果

幅 100・奥行 80・角 R12 → 円 R30、H90、偏心 X20／Y－10

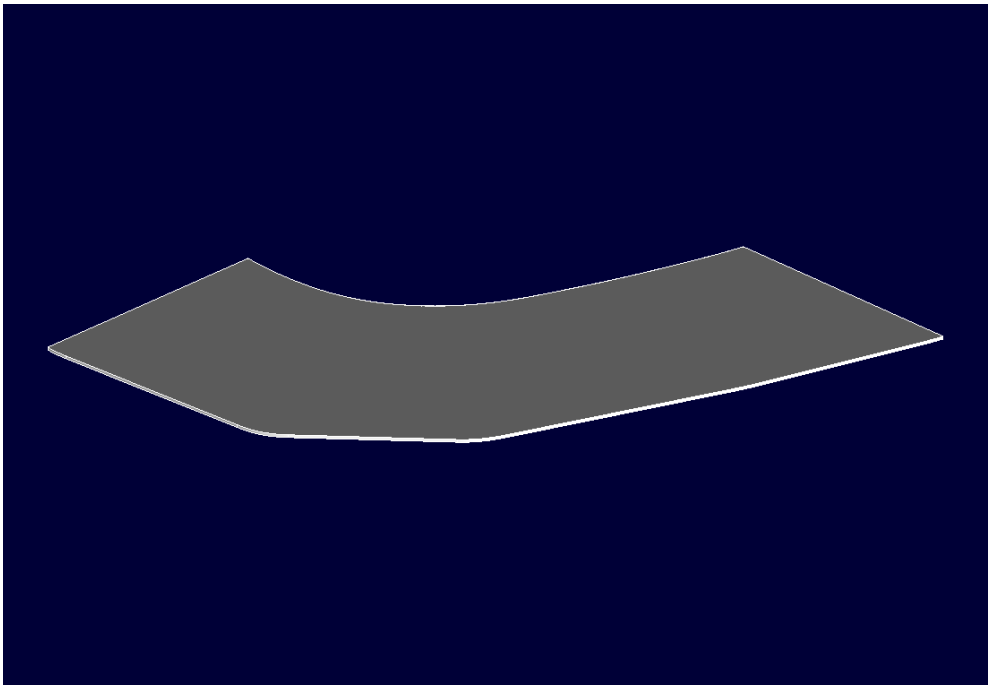


図 120　同じ角丸の実際の展開結果

同じ部品を展開した状態。輪郭は直線と滑らかな曲線です。

寸法とスケッチは中立面の基準です。角 R は矩形断面の R で、曲げ部の最小内 R とは異なります。実際の内外面は中立面から法線方向へ－K×板厚／＋(1－K)×板厚ずれます。偏心・傾斜した面の板厚端は、スケッチの平面と一致しません。

［切断位置：自動］では、継ぎ目は矩形の＋X・＋Y 側の角 R 中央に設けます。隙間は角 R と円半径の小さい方の中立面上の端点間距離です。大きい方の隙間は半径の比で広がります。隙間は両方の R より小さくしてください。［切断位置：指定］の手順と隙間の基準は下記をご覧ください。内寸・外寸基準の断面指定は対象外です。

［三角法］でも角 R を入力できます。曲面は分割面に近似されます。従来の角が直角の三角法を使う場合は角 R を 0 にします。既存の未加工部品は［既存ロール板を編集］から角 R と成形方法を変更できます。

角 R 付きの角―角を滑らかに作る

- ［断面の指定方法：寸法を入力］、［断面の種類：角→角（角 R・偏心対応)］を選びます。
- ［成形方法：滑らかな成形（ロール・型成形)］を選びます。
- 上下それぞれの幅・奥行き・矩形角 R と、高さを指定します。上側と下側の角 R は異なる値でも構いません。
- 偏心する場合は、上側中心のずれ X／Y を指定します。板厚、K 係数、継ぎ目の隙間を指定し［適用］します。
- ［板金展開］と［板金再曲げ］で切り替えます。［既存ロール板を編集］で上下の角 R や断面寸法を変更できます。

四隅の曲面が平らな側面に接線でつながります。上下同じ寸法の角筒、上側が大きい継手、偏心した継手にも対応します。各角 R は 0 より大きく、その矩形の幅・奥行きの半分未満にしてください。中立面寸法を使用し、内外面は板厚方向へオフセットします。

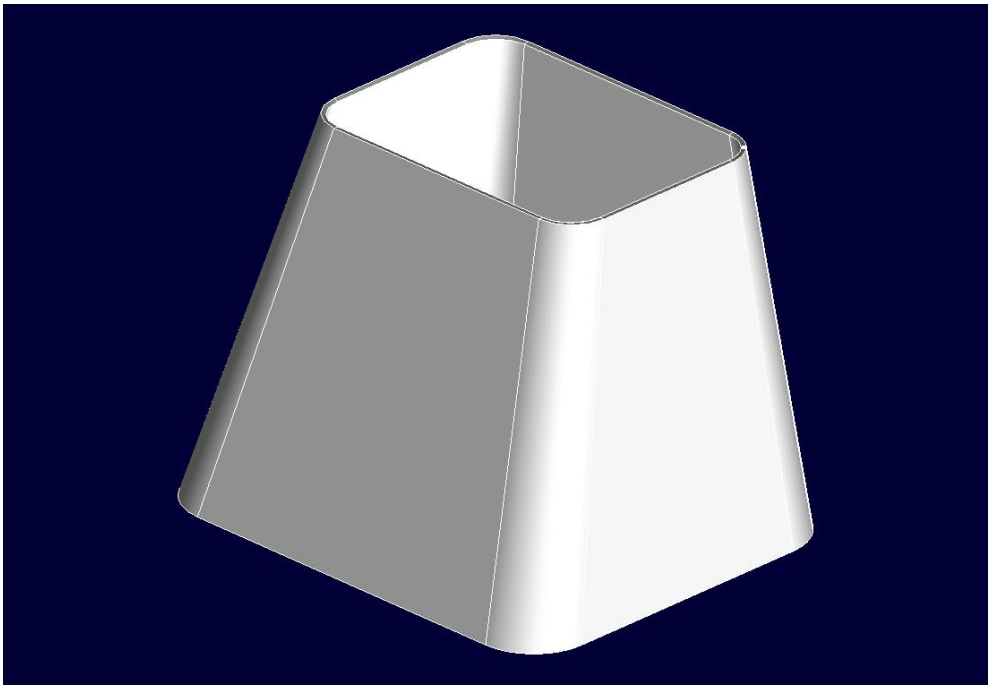


図 121　上下の角 R が異なる偏心角角の実形状

下側 100×80・R12 → 上側 70×50・R8、H90、偏心 X20／Y－10

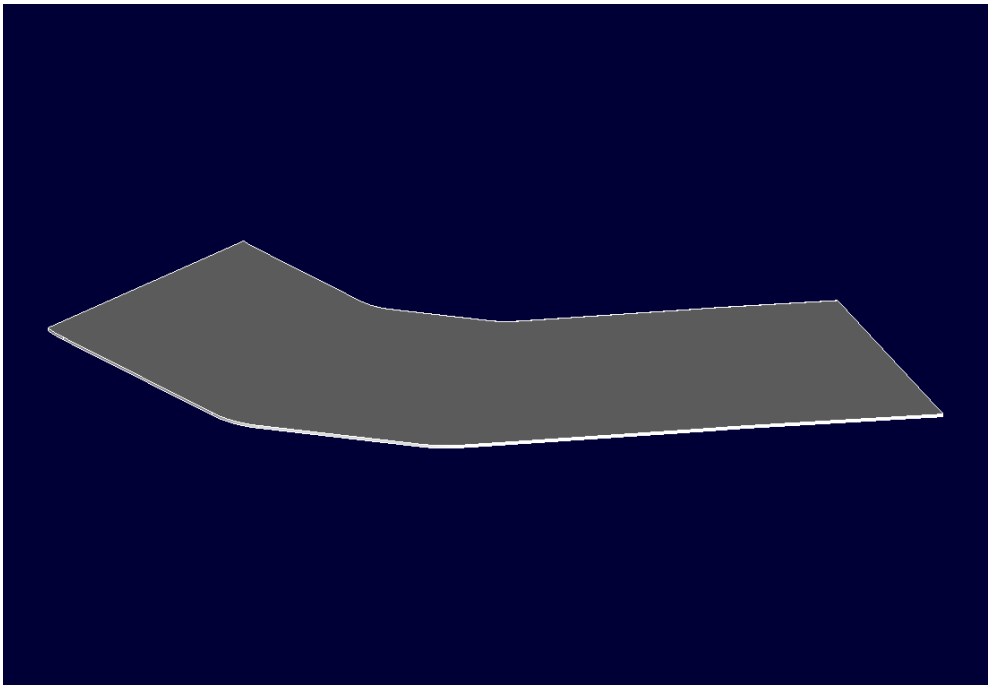


図 122　同じ角角継手の展開形状

同じ部品を展開。分割線を付けず、直線と曲線の輪郭で出力します。

2 枚の矩形スケッチからも作成できます。平行な平面上に、対応する辺が平行になるように描き、[2 つのスケッチを選択］で断面 1、断面 2 の順に選びます。各スケッチに角 R があればその値を使います。直角の矩形には、断面 1 は［下側の矩形角 R]、断面 2 は［上側の矩形角 R］を付けます。作成後にスケッチの寸法変更を自動反映する機能はありません。

［切断位置：自動］では、継ぎ目は局所座標の＋X・＋Y 側の角中央です。隙間は上下の小さい角 R 側の端点間距離で、もう一方は R の比率で広がります。両角 R より小さい隙間を指定してください。スケッチ指定時の局

所軸は断面 1 の直線から決まります。切断位置は下記の手順で指定できます。非平行断面、断面同士をねじった形は未対応です。

［三角法］でも上下の角 R を指定できます。旧来の直角の角一角は、三角法で両方の角 R を 0 にします。展開状態での貫通穴・切り欠き加工は再曲げできます。下記の加工手順をご覧ください。

2 枚のスケッチから作る

• 平行な 2 平面上に、矩形と円、円 2 つ、または矩形 2 つを描きます。矩形同士は対応する辺を平行にします。断面を横向きに配置しても構いません。

• ［ロール・ロフト］→［新規作成］→［断面の指定方法：2 つのスケッチを選択］を選びます。

• ［断面 1］［断面 2］で各輪郭を選びます。つながった直線・円弧は、その一部をクリックして選べます。矩形同士は断面 1 が下側、断面 2 が上側になります。同じ要素を再選択すると解除します。

• 成形方法、板厚、K 係数、継ぎ目の隙間を指定し、［適用］します。位置・高さ・偏心量・断面寸法はスケッチから読み取ります。

角 R 付き矩形はスケッチに描かれた R を使用します。直角の矩形なら、ダイアログの矩形角 R を付けます。矩形同士では下側・上側を個別に指定します。円 2 つでは角 R 入力を使いません。元のスケッチは残ります。作成後のスケッチ変更には連動しません。既存部品の編集では読み込んだ寸法を変更します。

矩形の輪郭は 4 直線、または同一 R の 90 度円弧 4 本と接線 4 本です。2 断面の平面は平行にしてください。矩形同士は対応する辺も平行にします。任意のスプライン輪郭、非平行断面、複数の中間断面には未対応です。展開後の貫通穴・切り欠き加工と再曲げについては、下記をご覧ください。

1. 種類を選ぶ

断面の種類	使用する形状	寸法基準
円筒・円すい（滑らかなロール）	同軸の円筒・円すい。従来の解析的な展開です。	上下の内半径・内側の軸方向高さ
丸→丸（偏心可）	円形から円形。中心がずれた継手にも対応。	上下の中立面半径
角→丸（角 R・偏心対応）	下側が角 R 付き長方形、上側が円形の角丸継手。滑らかな成形・三角法を選択。	下側の中立面幅・奥行・角 R、上側の中立面半径
角→角（角 R・偏心対応）	角 R 付き長方形どうし。滑らかな成形・三角法を選択。	上下の中立面幅・奥行・角 R

2. 作成手順

• ［操作：新規作成］と［断面の種類］を選びます。

• 配置位置は下側断面の中心です。断面はワールド XY に平行、高さは＋Z 方向です。作成後は通常の移動・回転・ミラーを使えます。

• 上下の断面寸法と高さ H を入力します。［上側中心のずれ X／Y］は下側中心に対する移動量です。負の値も使えます。

• 三角法の場合は［継ぎ目の隙間］と［周方向の分割数］を設定します。分割数は 8〜128 の 8 の倍数です。四角の角を必ず含め、各区間を 2 三角形に分けます。

• 板厚 t と K 係数を指定し、［適用］します。入力中には形状計算しません。

ロフトの入力寸法は中立面を表します。内側表面は中立面から −K×t、外側表面は +(1−K)×t の面直角距離です。入力寸法をそのまま内寸・外寸と解釈しないでください。三角形の接合部は面の交点で接続するため、板厚を付けた端の輪郭は中立面の断面輪郭と一致しません。

3. 三角法の継ぎ目と精度

［切断位置：自動］の角 R を使わない三角法では、継ぎ目は各断面の +X 側中央に設けます。隙間は上下それぞれの中立面上の端点間距離です。角 R 付きの角―丸・角―角では、上記の角中央の継ぎ目と、小さい R を基準とする隙間を使います。最初の分割点を越える隙間は指定できません。［切断位置：指定］では下側断面の方向を角度または CAD 上の点で指定します。複数の継ぎ目、断面のねじれは未対応です。

丸い断面は直線の弦で近似します。最大弦誤差は $R\times(1-\cos(\pi/N))$ です。画面の［三角法の近似精度］に表示します。例えば半径 40 mm、32 分割では約 0.193 mm、64 分割では約 0.0482 mm です。これは円形断面の誤差であり、継手全体の滑らかな曲面への誤差や成形精度を保証する値ではありません。

展開して穴・切り欠きを加工し、再曲げする

• ロフト板金を選び［板金展開］します。

• 通常の［穴］コマンド、またはスケッチからの［押し出し：切り取り］などで加工します。板厚を貫通する穴・長穴・外周の切り欠きにしてください。

• 加工後のソリッドを選び［板金再曲げ］します。平面と角 R の曲面をまたぐ長穴も、元の面に沿って戻ります。

• 再び［板金展開］すると加工輪郭が戻ります。追加の穴加工、保存・再読込後の再展開にも対応します。展開データの DXF／DWG には穴と切り欠きも出力されます。

滑らかな丸―丸（同心・偏心）、角―丸、角―角、三角法のロフト、および円筒・円すいロールが対象です。曲面での穴の形は、展開輪郭を材料の中立面に沿って戻した形になります。平板上の丸穴が、曲げ後にも同じ平面円になるとは限りません。

止まり穴、板厚の変わるポケットやテーパー、部品を分断する切断、元の展開材料より外側へ盛る加工には対応しません。加工後のロフト寸法・角 R は、下記の手順で変更できます。曲げたままの通常加工からの再展開は対象外です。エラーの場合は元の展開ソリッドを残します。

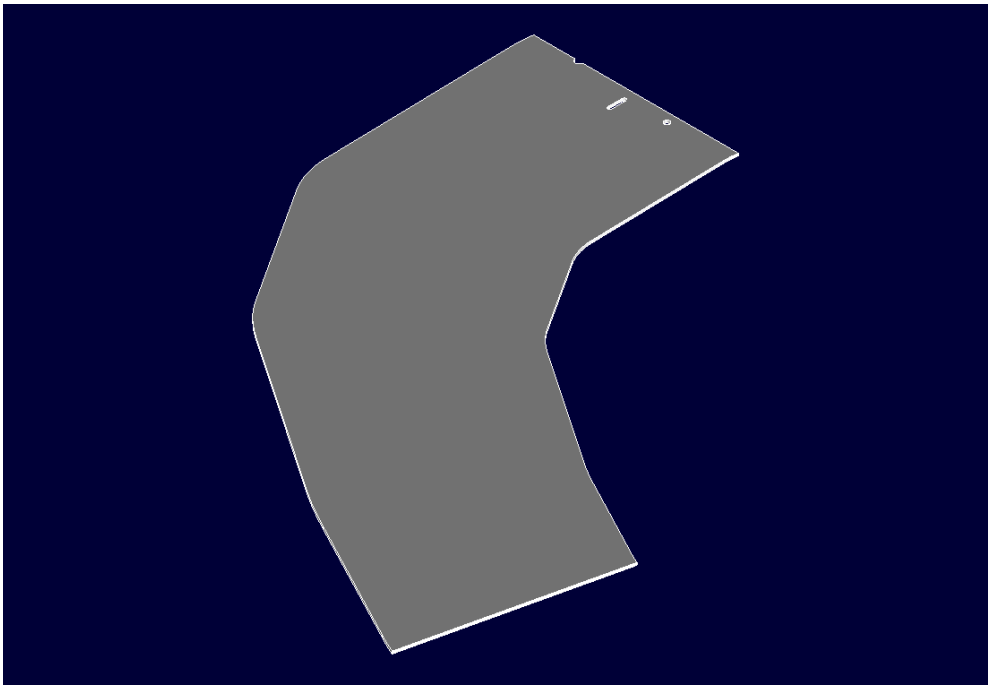


図 123　角角ロフトを展開し、穴・境界をまたぐ長穴・切り欠きを加工

通常コマンドで加工した展開板

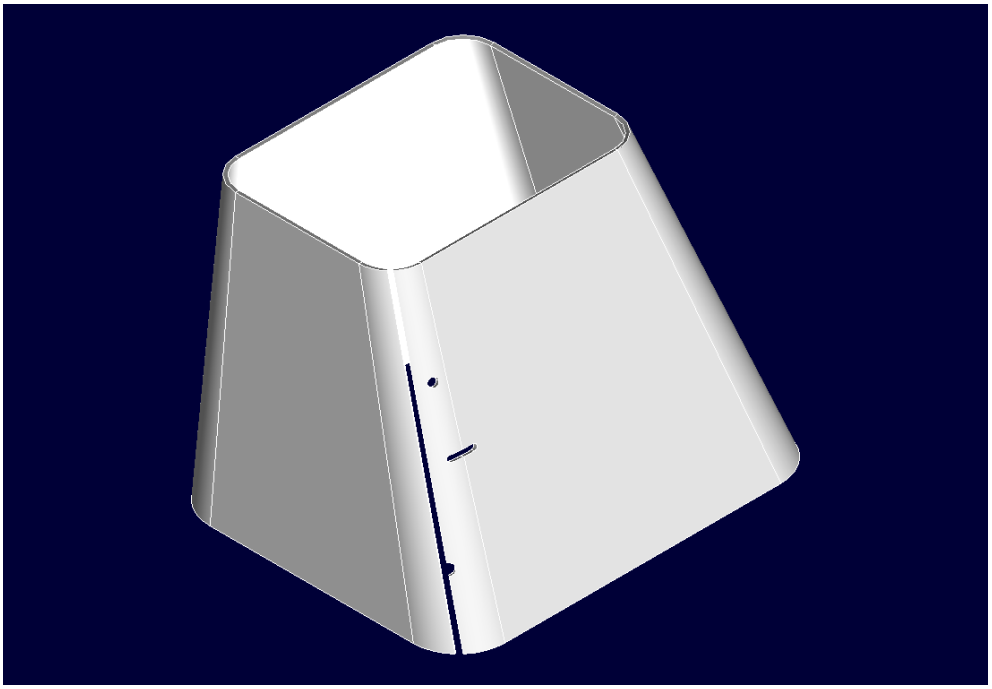


図 124　穴と長穴と切り欠きを保持して再曲げした角角ロフト

加工輪郭を保持して再曲げ

4. 展開・再曲げ・寸法編集

• 作成したソリッドに［板金展開］を適用します。三角法では境界エッジを残した平板、滑らかな成形では分割線のない平板になります。

• ［板金再曲げ］で元の継手へ戻せます。

• 寸法を変更するときは［ロール・ロフト］→［既存ロール板を編集］でソリッドを選びます。寸法・偏心量・隙間・分割数を読み込み、変更して［適用］します。展開状態のままの寸法編集にも対応します。

展開状態で通常コマンドにより作成した貫通穴・長穴・切り欠きを保持して、再曲げ・再展開できます。曲げたままの通常加工、止まり穴、通常コマンドによる不均一な板厚加工、元の材料範囲の外への追加は対象外です。加工後の寸法・角 R 編集では、下記の加工位置の基準を使います。有限 R の曲げ編集、エッジフランジ追加、バーリング等の成形追加、部分展開の対象にもなりません。ロフト寸法と板厚の編集、部品全体の移動・回転・ミラーを使用してください。

加工を残して寸法・角 R を変更する

• ［ロール・ロフト］→［既存ロール板を編集］で、加工済みのソリッドを選びます。展開中でも再曲げ後でも操作できます。

• 上下の断面寸法、高さ、偏心量、角 R などを変更し、［適用］します。角一角は上下の R を別々に変更できます。

• ［板金展開］で加工位置を確認します。［元に戻す］で寸法と加工の両方を変更前に戻せます。

加工位置は展開座標で固定します。ロフト本体のローカル展開 XY 座標を基準に、穴・長穴・切り欠きの加工輪郭を移動・回転・拡大縮小せず保持します。面の大きさに比例した位置移動や、端からの一定距離への自動追従は行いません。部品全体を移動・回転・ミラーした場合は、その座標系も一緒に動きます。

穴径・長穴輪郭はそのままです。外周を動かすと、固定された切り欠きと外周の接し方が変わります。材料が広がって加工領域を囲めば穴になります。変更前の切り欠きで実際に除去した領域を保持する方式で、元の押し出し工具の板外部分を推測して延長することはありません。連続した寸法変更では加工領域を保存するため、元の寸法に戻せます。通常コマンドで追加加工した時点で、その時の実形状から加工輪郭を取り直します。

閉じた穴が外周に掛かる、加工領域が完全に板の外へ出る、材料が分断されるなど、加工を保持できない寸法では適用を中止します。元のソリッドは残ります。加工位置を調整する場合は展開して通常コマンドで加工し直してください。新しい加工情報を保存した JDC は、この版以降で編集してください。

5. 展開図・DXF／DWG

［展開図 作成・更新］または［展開データ出力］を使います。三角法の［曲げ線：表示］では内部基準線を出します。滑らかな成形ではこの基準線を出しません。この線は切断線や曲げ R を持つプレス曲げ線ではありません。

レイヤー	内容
SM_CUT	展開した材料の外周
SM_LOFT_GUIDE	三角法の内部基準線。対角線も含みます。
SM_LOFT_NOTE	分割数・弦誤差・プレス曲げ補正を含まない旨の注記

基準線は［曲げ線：非表示］、注記は［注記：非表示］で外せます。DXF と DWG は同じ区分で出力します。曲げ一覧表の B 番号は付けません。板金チェックの最小曲げ R 検査は、有限 R を持たない三角法ロフトでは使用できません。

6. 入力例

形状	下側	上側	高さ・偏心・隙間
偏心の丸丸	半径 40	半径 25	H80、X20、Y10、隙間 2
角丸	幅 90、奥行 65	半径 25	H80、X20、Y10、隙間 2
角角	幅 90、奥行 65	幅 50、奥行 40	H80、X20、Y10、隙間 2

寸法単位は mm、板厚 1、K＝0.4、分割数 32 の例です。極端な偏心、短すぎる高さ、厚すぎる板、展開が重なる条件は作成できません。新しいロフト情報を含む JDC は、この版以降で編集してください。

参考にした公式資料

• Fusion: lofted flange, die forming and brake forming

• Inventor: lofted flange types

第 8 部　カスタマイズガイド

自社用の作図・集計コマンドを追加するための章です。Free 版では配布時の API サンプルのみを実行します。独自 DLL・改造したサンプルは STD／Pro で利用してください。MCP の接続は Free 版でも利用できます。

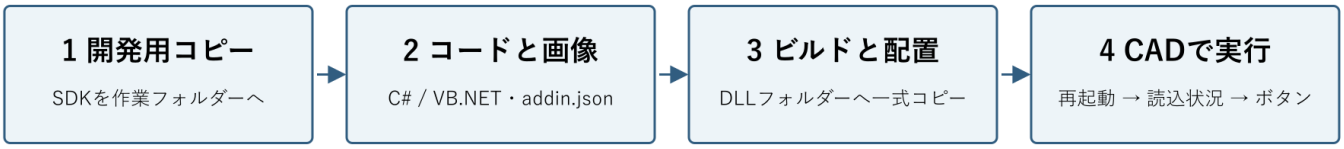


図 125　DLL を作って使うまでの順序。



図 126　付属 API サンプルの 10 コマンド。



図 127　［読込状況］で結果を確認し、［DLL フォルダー］で配置先を開きます。

カスタマイズの配置と利用条件

公開メソッドリファレンス：宣言・引数・戻り値・使用例

Free 版とカスタマイズの利用条件

Free 版は配布時の「API サンプル」10 コマンドだけを利用できます。独自 DLL と変更したサンプルの実行には、有効な STD または Pro ライセンスが必要です。Free 版ではサンプルの名前・ID を使っても外部 DLL のコードは実行せず、本体に同梱されたサンプルを実行します。変更が反映されない場合は［読込状況］とプランを確認してください。C#／VB.NET の実習サンプルを自作 DLL として読み込む場合も STD／Pro が必要です。

MCP は Free 版でも使用できます。通常の作図・編集 API は利用でき、板金・解析の API はライセンス判定で拒否されます。独自 DLL の読込み制限と MCP 接続の可否は別です。

JDesignCAD の追加 DLL、独立した.NET プログラム、MCP は同じ共通 API を使用します。図形 ID・座標・寸法などを明示して呼び出し、図面への反映と Undo は本体が管理します。API に CAD エンジンは含まれません。対象の JDesignCAD と図面を開いて使います。

操作の 3 つの形

種別	用途	完了の判断
数値・ID 指定の自動操作	作図、図形定義・属性・レイヤー変更、変換、ブロック、寸法	ExecuteOperationAsync が EditResult を返す。図形変更は 1 回の Undo。
別プロセスで計算	3D 加工、板金、曲線加工、計測、解析	ready なら CommitJobAsync で図面へ反映。計測・解析は completed になり、GetJobDataAsync で結果を読む。
対話コマンドの起動	既存の作図入力、印刷・保存画面、設定など	queued は起動予約の受付。ユーザーが CAD 画面で完了する。処理完了の通知ではない。

「すべてのコマンドを起動できること」と「すべてを無人で実行できること」は別です。API は 75 種類の構造化操作と、266 個の登録コマンドの一覧・対話起動を提供します。ブール演算 3 件は別プロセスの API を使います。形状依存の失敗、画面操作が必要な機能、未公開の編集項目は対応表に記載しています。板金・解析は本体と同じ STD／Pro の条件が適用されます。

配布物と DLL の配置

```
JDesignCAD.exe のフォルダー/
  JDesignCAD.Api.dll                    公開契約
  JDesignCAD.Automation.Worker.exe      計算用ワーカー
  Customization/
    使い方.html                         このマニュアル
    reference.ja.html                   API リファレンス
    coverage.ja.html                    対応表
    SDK/Api/                            DLL と IntelliSense 用 XML
    SDK/Samples/Customization.Sample/   実用コマンド 10 例
```

```
SDK/Samples/Customization.Workshop/   C#実習
SDK/Samples/Customization.WorkshopVB/ VB.NET 実習
Mcp/JDesignCAD.Mcp.exe
mcp-config.json
```

追加 DLL の配置先は%LOCALAPPDATA%¥JDesignCAD¥Addins¥拡張名です。［カスタマイズ］→［DLL フォルダー］から開けます。各フォルダーへ addin.json、作成した DLL、依存 DLL、アイコンを配置し、JDesignCAD を再起動します。

DLL は本体と同じ Windows ユーザーの権限で動作します。読込分離はセキュリティ用の隔離ではありません。自作または内容を確認した拡張を配置してください。

API サンプル：実用コマンド

［API サンプル］タブには文字連番・置換・大文字小文字・文字囲み、等間隔複写・同種同層選択・重複線整理・色を ByLayer、長さ面積集計・ブロック集計の 10 例を配置しています。共通 API だけで作るサンプルです。操作方法・対応 API・ソースの読み方をご覧ください。［カスタマイズ］タブの管理コマンドは変更していません。

API の共通ルール

• コマンドやダイアログを終了し、モデル空間をアクティブにします。

• GetDocumentAsync で図面 ID と revision を取得します。

• 図形は選択・一覧取得で返された ID を使います。ID は現在開いている図面のセッション内だけで有効です。

• 変更には新しい UUID の operationId を付けます。通信切断後の再試行は同じリクエスト・同じ UUID にします。

• 変更後は返された DocumentInfo を使うか、再取得します。面・辺・頂点の参照は revision が変わるたびに取り直します。

座標は WCS、長さは mm、角度は度です。表示単位・UCS に依存しません。ロックしたレイヤー、Undo 無効、他のコマンド実行中、古い revision には書込みを拒否します。大きな図面では一覧と結果をページ単位で取得してください。

問題が起きたとき

症状	確認
タブが出ない	［読込状況］。addin.json の assembly・entryType・enabled、.NET 8/x64、DLL 参照。
以前の DLL が動く	本体を終了して差し替え、再起動。重複した addin id や配置先違いも確認。
cad_busy	CAD 入力・ダイアログを終了。UI スレッドで Task.Wait／Result を使わず Await／await する。
revision_conflict	図面・対象・面参照を再取得して計算をやり直す。古い計算結果に新 revision を付けても反映不可。

症状	確認
worker_unavailable	本体と同じ版の Automation.Worker の exe・dll・deps.json・runtimeconfig.json を一緒に配布。
geometry_failed / invalid_result	寸法、板厚、閉鎖、交差、辺・面の参照を確認。元図面は保持。全形状で成功する保証はない。
license_required	STD／Pro が必要。DLL と MCP で同じ判定。
message_too_large	ローカル通信は 1 メッセージ 1 MiB まで。取得ページを小さくする。巨大な解析入力は MCP 送信に制約あり。

読込・対話起動ログは%LOCALAPPDATA%¥JDesignCAD¥Automation 配下です。queued 後に画面が変わらない場合は、現在の選択・画面・コマンドの有効状態も確認します。

プログラム作成とデバッグ

Free 版とカスタマイズの利用条件

Free 版は配布時の「API サンプル」10 コマンドだけを利用できます。独自 DLL と変更したサンプルの実行には、有効な STD または Pro ライセンスが必要です。Free 版ではサンプルの名前・ID を使っても外部 DLL のコードは実行せず、本体に同梱されたサンプルを実行します。変更が反映されない場合は［読込状況］とプランを確認してください。C#／VB.NET の実習サンプルを自作 DLL として読み込む場合も STD／Pro が必要です。

MCP は Free 版でも使用できます。通常の作図・編集 API は利用でき、板金・解析の API はライセンス判定で拒否されます。独自 DLL の読込み制限と MCP 接続の可否は別です。

1. 開発環境

• Visual Studio 2022 17.8 以降と.NET 8 SDK を用意します。Visual Studio Installer で［.NET デスクトップ開発］を選択します。

• 更新版 JDesignCAD を用意します。SDK は本体フォルダーの Customization¥SDK にあります。

• SDK フォルダー全体を開発フォルダーへコピーします。Api と Samples の相対位置を維持すると、同梱プロジェクトの参照がそのまま使えます。

Microsoft 公式：Visual Studio でクラスライブラリを作る、.NET と Visual Studio の対応。

2. 同梱サンプルから開始する

Visual Studio の［プロジェクトまたはソリューションを開く］で、次のいずれかを開きます。

実用コマンド 10 例を改造する場合は API サンプルの説明をご覧ください。Windows Forms 入力画面を使うため UseWindowsForms=true が必要です。

言語	ファイル	内容
C#	SDK¥Samples¥Customization.Sample¥Customization.Sample.csproj	API サンプル：文字連番・複写・整理・集計の 10 例。API 1.3 以降。
C#	SDK¥Samples¥Customization.Workshop¥Customization.Workshop.csproj	タブ、ボタン、線と円、箱、滑らかな角一角ロフト、選択情報、Undo
VB.NET	SDK¥Samples¥Customization.WorkshopVB¥Customization.WorkshopVB.vbproj	タブ、ボタン、線分作成。Option Strict On。

構成は Release、プラットフォームターゲットは x64 です。［ビルド］→［ソリューションのビルド］を実行します。通常の出力先は bin¥Release¥net8.0-windows です。クラスライブラリのため、DLL を F5 で直接実行することはできません。

3. 新規プロジェクトから開始する場合

• ［新しいプロジェクトの作成］→C#または Visual Basic のクラスライブラリを選択します。.NET Framework 版ではなく、.NET 8 版を使用します。

• プロジェクトファイルを次のように設定します。HintPath は実際の SDK に合わせます。

• ［依存関係］→［参照の追加］→［参照］で SDK¥Api¥JDesignCAD.Api.dll を指定します。参照の［ローカルにコピー］は False にします。

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <TargetFramework>net8.0-windows</TargetFramework>
    <PlatformTarget>x64</PlatformTarget>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
    <EnableDynamicLoading>true</EnableDynamicLoading>
  </PropertyGroup>
  <ItemGroup>
    <Reference Include="JDesignCAD.Api">
      <HintPath>..¥..¥Api¥JDesignCAD.Api.dll</HintPath>
      <Private>false</Private>
    </Reference>
    <None Update="addin.json" CopyToOutputDirectory="PreserveNewest" />
  </ItemGroup>
</Project>
```

WinForms の入力画面を作る場合は<UseWindowsForms>true</UseWindowsForms>も追加します。VB.NET では RootNamespace と OptionStrict を設定します。内部の JDesignCore や Krypton の参照は、タブ・ボタン登録や共通 API の利用には不要です。

EnableDynamicLoading と共有契約 DLL について：Microsoft のプラグイン対応アプリの説明。

4. 起動クラスとコマンドを書く

ICadAddin.OnStartup でタブを登録し、各ボタンには IAddinCommand を実装したクラスを渡します。OnStartup は登録だけを行い、図面変更や長い処理はボタンから実行します。コードは同梱の WorkshopAddin.cs／WorkshopAddin.vb と実装例を参照してください。

非同期処理は await／Await で呼び出します。UI スレッドで.Result や.Wait()を使うと、UI へ戻る処理を待ち続ける可能性があります。重い 3D 計算は StartOperationAsync を使用し、完了後に CommitJobAsync を呼びます。

5. addin.json を作る

```
{
  "id": "MyCompany.MyTool",
  "name": "自社の加工支援",
  "apiVersion": 1,
  "assembly": "MyTool.dll",
  "entryType": "MyCompany.MyTool.MyAddin",
  "enabled": true
}
```

項目	設定
id	ほかの拡張と重複しない識別子。同じ拡張の更新では維持。
apiVersion	主バージョンの 1。API 1.2・1.3 でも 1 です。

項目	設定
assembly	ビルドで生成された DLL 名。拡張フォルダー内の相対パス。
entryType	ICadAddin を実装した public クラスの完全名。引数なしで生成できること。
enabled	false で読込みを無効化。

VB.NET の完全名は「プロジェクトのルート名前空間＋クラス名」です。ソースにも同じ Namespace を書くと二重になるため、同梱 VB サンプルではソース側の Namespace を省いています。

6. 配置して起動

• 図面を保存して JDesignCAD を終了します。

• %LOCALAPPDATA%¥JDesignCAD¥Addins¥MyCompany.MyTool を作成します。

• 自分の DLL、addin.json、生成された deps.json、使用している依存 DLL、必要なら pdb・アイコンをコピーします。JDesignCAD.Api.dll は本体側のものを共有します。

• JDesignCAD を再起動し、［カスタマイズ］→［読込状況］を確認します。

• 図面を開いて追加タブのボタンを押します。

SmallIconPath／LargeIconPath は拡張フォルダーを基準としたパスです。未指定は画像なし。グループを折りたたんだときは、先頭ボタンのアイコンが使われます。DLL 更新後は本体の再起動が必要です。

7. ブレークポイントで確認

• Debug 構成でビルドし、DLL と pdb を拡張フォルダーへ配置して JDesignCAD を起動します。

• Visual Studio の［デバッグ］→［プロセスにアタッチ］で JDesignCAD.exe を選び、マネージド（.NET Core／.NET 5 以降）コードへ接続します。

• ExecuteAsync の先頭へブレークポイントを置き、CAD のボタンを押します。

• 停止しないときは［デバッグ］→［ウィンドウ］→［モジュール］で、読み込んだ DLL の場所とシンボルを確認します。

OnStartup を追うときは、デバッグ設定で外部プログラムとして JDesignCAD.exe を指定して起動するか、起動時ログを利用します。計算エンジンは別の Automation.Worker プロセスです。追加 DLL 側ではジョブの state・errorCode・message を確認してください。Microsoft 公式：実行中プロセスへのアタッチ。

8. 配布と互換性

拡張フォルダーをまとめて配布します。実習サンプルは API 1.2 以降、板金・解析は STD／Pro が必要です。SDK の公開 DLL と XML を一緒に参照すると IntelliSense に説明が出ます。内部の CAD 型や JDC 構造を直接書き換える実装は、共通 API の互換性保証に含まれません。

付属サンプルの使い方

Free 版とカスタマイズの利用条件

Free 版は配布時の「API サンプル」10 コマンドだけを利用できます。独自 DLL と変更したサンプルの実行には、有効な STD または Pro ライセンスが必要です。Free 版ではサンプルの名前・ID を使っても外部 DLL のコードは実行せず、本体に同梱されたサンプルを実行します。変更が反映されない場合は［読込状況］とプランを確認してください。C#／VB.NET の実習サンプルを自作 DLL として読み込む場合も STD／Pro が必要です。

MCP は Free 版でも使用できます。通常の作図・編集 API は利用でき、板金・解析の API はライセンス判定で拒否されます。独自 DLL の読込み制限と MCP 接続の可否は別です。

［API サンプル］タブの 10 コマンドは、公開 API だけで作れるカスタマイズの例です。文字処理、複写、選択、整理、集計を組み合わせています。［カスタマイズ］タブの［読込状況］［DLL フォルダー］は従来どおりです。



図 128　API サンプルの文字・複写整理・集計の 3 グループと 10 ボタン

サンプル DLL を実際に読み込んだリボンの確認画面。

用途の参考にしたのは AutoCAD Express Tools のコマンド一覧、TCOUNT、COPYM です。JDesignCAD 向けに独自実装したサンプルで、Express Tools とのコマンド・ファイル・動作互換を提供するものではありません。

使い方

• モデル空間で対象を選択します。文字コマンドは通常の文字、同種・同層選択は基準図形 1 個を選びます。ブロック集計には選択は不要です。

• ［API サンプル］のコマンドを押し、距離や文字を入力します。文字編集は変更内容、重複線整理は削除件数の確認後に適用します。

• 図形を変更するコマンドは 1 回の［元に戻す］で取り消せます。選択と集計は図形を変更しません。

単位は WCS・mm です。UCS や表示単位を変更していても同じです。図面が処理中に変更された場合は古い入力を反映せず、選択し直して再実行します。

10 個のコマンドと API

コマンド	動作と範囲	公開 API
文字連番	開始番号・増分・前後の文字・X 昇順／Y 降順／API 一覧順・全体置換／先頭追加／末尾追加。例：開始 5、増分 3、前の文字 P-で P-5、P-8…。クリックした順序の保証はありません。	QueryAsync(geometry) → text.update

コマンド	動作と範囲	公開 API
文字置換	選択文字内の一致部分を全て置換。大文字・小文字を区別。正規表現は使いません。変更後の文字全体が空になる置換は拒否します。	QueryAsync(geometry) → text.update
大文字・小文字	英字などを大文字／小文字に変更。通常の日本語は変化しません。変更のない文字には Undo を追加しません。	QueryAsync(geometry) → text.update
文字囲み	文字ごとに矩形または円を追加。WCS XY に平行な文字のみ。文字の実際の外接範囲に余白を足し、円はその矩形を囲みます。囲みは文字と同じ Z、色・レイヤーは現在の作図設定です。傾斜した 3D 文字、追従する関連付けは対象外。	EntityInfo.Bounds → geometry.create
等間隔複写	X・Y・Z の間隔と複写数を入力。元の図形は残し、1 倍、2 倍…の位置へ複写。複写数は元を含みません。生成図形は合計 1000 個まで。	entities.array.linear
同種・同層選択	基準の 1 図形と同じ Kind・レイヤーの図形をモデル直下から選択。ブロック内部は検索しません。色は条件に含めません。検索 10 万図形、結果 1000 個まで。	GetEntitiesAsync → SetSelectionAsync
重複線整理	選択した線分・円が全体として重なり、レイヤー・色指定・線種・線幅が同じ場合に 1 つを残す。既定許容差 0.001 mm。反対向きの線も判定。ロック中、部分重複、円弧・ポリラインは対象外。任意の拡張メタデータは比較していません。	GetEntityDetailsAsync + GetAttributesAsync → DeleteEntitiesAsync
色を ByLayer	選択図形の色だけをレイヤーに合わせる。レイヤー・線種・線幅は変更しません。ロックされた対象が含まれる場合は全体を変更せず終了します。	attributes.set
長さ・面積集計	選択曲線の長さと閉じた平面曲線の面積を集計。100 図形まで。mm／m、mm²／m²を表示。重なりや入れ子の穴は差し引かず、個別面積を加算。文字・ソリッド等は長さの対象外として数えます。計算中に中止できます。	StartOperationAsync(measure.geometry) → GetJobDataAsync

コマンド	動作と範囲	公開 API
ブロック集計	アクティブモデル直下に直接挿入した名前付きブロックを名前別に集計。入れ子の内部、紙空間、無名ブロックは除外。定義 1000 件に達した場合は結果が一部の可能性を表示。	QueryAsync(blocks)

文字処理は選択中の通常文字 1〜100 個、全コマンドの選択取得は 1000 図形までです。ブロック内文字・属性定義・寸法文字の編集例ではありません。text.update はフォント、太字・斜体、位置、配置基準、文字高さ、向き、色・層、ID を保持します。

Visual Studio で変更する

• 本体フォルダーの Customization¥SDK 全体を開発用フォルダーにコピーし、Samples¥Customization.Sample¥Customization.Sample.csproj を開きます。

• ターゲットは net8.0-windows、UseWindowsForms=true です。公開 DLL の参照先は../../Api/JDesignCAD.Api.dll、ローカルにコピーは False です。

• Release でビルドします。ソースを読んでボタン名やパラメータ、処理を変更できます。別の拡張として追加する場合は addin.json の id と name を変更してください。同じサンプルを更新する場合は id を維持します。

• 本体を終了し、出力 DLL・addin.json・20 枚の PNG を%LOCALAPPDATA%¥JDesignCAD¥Addins¥JDesignCAD.Customization.Sample に配置して再起動します。変更前のフォルダーは Addins の外へ保存してください。

ソース	読む内容
SampleAddin.cs	タブと 3 グループ、10 ボタン、アイコンの登録。タブ名は API サンプル。
ToolCommand.cs	選択取得、入力、revision 付き要求、API の組み合わせ、Undo 単位、ジョブの中止。
ToolGeometry.cs	公開 DTO だけで行う重複判定と計測結果の集計。
ToolDialogs.cs	WinForms 入力・確認・結果表示。IToolUi を差し替えると画面操作なしで検証可能。
Build-Icons.ps1 / *.png	独自の簡易ベクター図案から 24／48px の PNG を生成。ビルド済み PNG を同梱。

文字内容だけを更新する最小例

```
var selected = await api.GetSelectionAsync(cancellationToken: ct);
var cad = (ICadAutomationApi)api;
await cad.ExecuteOperationAsync(selected.Document.Operation(
    "text.update",
    new UpdateTextParameters([
```

```
        new TextUpdate(selected.Entities[0].Id, "P-001")
    ])), ct);
```

通常文字を 1 個選んだ場合の例です。実際のサンプルは空選択・件数・変更前内容を確認します。文字内容だけの編集には text.update を使います。geometry.update に TextDefinition を渡す方法は文字定義の置換となり、取得 DTO に含まれない書式まで保持する用途には向きません。

MCP でも同じ処理を呼ぶ

タブの入力ダイアログは DLL 側の UI です。MCP では選択取得・geometry 照会で対象 ID を読み、同じ公開操作に入力値を指定します。文字編集は cad_execute_operation の request.operation="text.update"、parameters は次の形です。documentId・expectedRevision・新しい operationId も必要です。

```
{
  "updates": [ { "entityId": "取得した文字 ID", "content": "P-001" } ]
}
```

複写は entities.array.linear、色変更は attributes.set、計測は cad_start_operation から measure.geometry を呼びます。サンプル固有の並べ替えや重複判定は呼出し側で行います。MCP からサンプルのダイアログを直接操作する仕様ではありません。

共通 API の実装例

以下は IAddinCommand.ExecuteAsync 内の C#コードです。api は ICadApi、ct は渡された CancellationToken を表します。コード全体が必要な場合は、同梱の Customization.Workshop プロジェクトを使用してください。

線分・楕円・文字をまとめて作成

```
if (api is not ICadAutomationApi cad)
    throw new InvalidOperationException("API 1.2 が必要です。");
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("geometry.create", new CreateObjectsParameters([
    new(Line: new(new(0, 0, 0), new(50, 0, 0))),
    new(Ellipse: new(new(25, 20, 0), new(15, 0, 0), new(0, 8, 0))),
    new(Text: new("加工基準", new(0, 40, 0), new(1, 0, 0), new(0, 1, 0), 5))
]));
var result = await cad.ExecuteOperationAsync(request, ct);
// 全図形をまとめて、1 回の Undo で取り消せます。
return new CommandResult($"{result.Entities.Length}個作成しました。");
```

GeometryDefinition は Line／Circle／Arc／Polyline／Ellipse／Spline／Text／Point のうち 1 つだけを指定します。Spline は通過点指定です。詳細取得のスプラインは正確な極点・節点・重みを返すため、そのまま通過点入力として使うことはできません。

選択図形を回転する

```
var cad = (ICadAutomationApi)api;
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
if (selection.Total == 0) return new CommandResult("図形を選択してください。");
if (selection.Total != selection.Entities.Length)
    return new CommandResult("ページを追加取得してから実行してください。");
var ids = selection.Entities.Select(e => e.Id).ToArray();
var request = selection.Document.Operation("entities.rotate",
    new RotateParameters(ids, new(0, 0, 0), new(0, 0, 1), 30));
await cad.ExecuteOperationAsync(request, ct);
return new CommandResult("WCS の Z 軸を中心に 30 度回転しました。");
```

箱の作成：計算と反映を分ける

```
var cad = (ICadJobApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("solid.primitive",
    new PrimitiveParameters("box", new(0, 0, 0), new(20, 30, 10)));
var job = await cad.StartOperationAsync(new(request, TimeoutSeconds: 120), ct);
try
{
    while (job.State == "running")
    {
        await Task.Delay(250, ct);
        job = await cad.GetJobAsync(job.Id, ct);
    }
    if (job.State != "ready")
        throw new InvalidOperationException(job.ErrorCode + ": " + job.Message);
    var current = await cad.GetDocumentAsync(ct);
    var committed = await cad.CommitJobAsync(new(current.Id, current.Revision,
        Guid.NewGuid().ToString("D"), job.Id), ct);
    return new CommandResult("箱を作成しました。");
}
catch
{
    try { await cad.CancelJobAsync(job.Id, CancellationToken.None); } catch { }
    throw;
}
```

ready になるまでは図面を変更しません。計算中に手動編集や別 API の変更が入ると、CommitJobAsync は拒否されます。無理に revision だけを書き換えず、対象を取り直して計算します。タイムアウト・中止は API が起動した計算プロセスを停止します。

面・辺を選んで加工

GetTopologyAsync で kind を face／edge／vertex に指定して取得します。返された Reference をそのまま FaceOffsetParameters／FilletParameters 等へ渡します。Index を固定値で決め打ちせず、Bounds、GeometryType、Position、Normal、Length と対象図形の情報を使って選びます。平面以外の面では Position／Normal が null になることがあります。

```
var doc = await cad.GetDocumentAsync(ct);
var faces = await cad.GetTopologyAsync(new(doc.Id, doc.Revision, solidId,
    Kind: "face", Limit: 256), ct);
// wantedFace は取得結果から条件で決定。複数候補なら利用者に選択してもらう。
var request = doc.Operation("face.offset",
    new FaceOffsetParameters(solidId, [wantedFace.Reference], 5));
// StartOperationAsync → ready → CommitJobAsync の順に実行。
```

面のコピー・移動で得た新しいソリッドには新しい ID が返ります。次の加工には結果の Entities を使用します。形状更新前の辺・面番号を使い回してはいけません。

板金の作成・加工・再編集

```
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("sheet.loft", new SheetLoftParameters(
    new SheetRules(Thickness: 1, Radius: 1, KFactor: .4),
    new SheetLoftDimensions(Kind: 3, FormingMode: 1, Height: 80,
        BottomWidth: 100, BottomDepth: 80, TopWidth: 60, TopDepth: 40,
        CornerRadius: 10, TopCornerRadius: 6,
        OffsetX: 10, SeamMode: 1, SeamAngle: 45)));
// ジョブで作成。STD/Pro が必要。
```

• sheet.unfold を実行し、返されたソリッド ID を取得します。

• 円・閉輪郭から geometry.extrude で工具ソリッドを作り、solid.boolean の subtract で穴・切欠きを加工します。Boolean は先頭 ID が加工される母材です。

• sheet.refold で加工を保持して再曲げします。

• sheet.loft の EntityId に現在のロフト ID を指定すると、Dimensions／Rules で寸法と角 R を変更できます。加工位置は展開板の基準からの距離を維持します。

既存の板金エンジンと同じ形状条件が適用されます。加工が新しい板の外側に出る、曲げとの干渉、非展開形状などは失敗します。自由なソリッド変形をすべて板金として復元する機能ではありません。

計測・解析の結果を読む

measure.geometry、analysis.mesh、analysis.solve は図形を挿入しません。state が completed になったら GetJobDataAsync を呼びます。CommitJobAsync は使いません。

```
var workspace = (ICadWorkspaceApi)api;
var summary = await workspace.GetJobDataAsync(new(jobId, "summary"), ct);
var firstPage = await workspace.GetJobDataAsync(new(jobId, "items", 0, 128), ct);
// measure.geometry: summary / items
// analysis.mesh: summary / points / elements / boundaries / edgeBoundaries / vertexNodes
// analysis.solve: summary / values / elementValues / nodeReactions /
//               elementStrains / equivalentStrains / thermalFrames
```

解析は単一材料を対象とします。メッシュの節点番号は 0 始まり、要素は 2 次要素です。荷重は N、応力は N/mm²、熱流束は W/mm²、対流係数は W/(mm² K)です。材料の熱伝導率 W/(m K)とは単位が異なるため、リファレンスを確認してください。

独立した.NET プログラムから接続

```
var sessions = LocalProtocol.FindSessions();
// 対象ウィンドウを利用者に選んでもらい、対応する SessionInfo を selected に設定。
ICadInteractionApi cad = new LocalCadClient(selected);
var doc = await cad.GetDocumentAsync();
```

同じ Windows ユーザーの起動中 JDesignCAD へ接続します。別の図面へ自動で切り替えることはありません。外部プログラムでも revision、operationId、ライセンスと Undo の規則は共通です。

付属サンプルをインストールする

図面を保存して CAD を閉じ、本体の Customization フォルダーにある Install-Sample.ps1 を PowerShell から実行します。既存フォルダーがある場合は上書きを停止するため、変更を保管して手動で更新してください。再起動後に［カスタマイズ］→［読込状況］を確認します。

```
powershell -File "C:¥Program Files¥JDesignCADV1¥Customization¥Install-Sample.ps1"
```

C#の実装全体

同梱 SDK のコードです。対応するプロジェクトと addin.json を使い、STD／Pro でビルドした DLL を読み込みます。行の折り返しは紙面上の表示です。

```
using JDesignCAD.Api;
namespace Customization.Workshop;

public sealed class WorkshopAddin : ICadAddin
{
    public void OnStartup(IAddinContext context)
    {
        context.RegisterTab(new("workshop","API 実習",
        [new("create","作成",[
            new("curves","線と円","1 回の Undo で取り消せる線と円",new WorkshopCommand("curves")),
            new("box","箱を作成","別プロセスで 20×30×10 mm の箱を計算",new WorkshopCommand("box")),
            new("loft","角–角ロフト","STD/Pro。滑らかな角–角ロフトを作成",new WorkshopCommand("loft"))]),
         new("inspect","確認",[
            new("selection","選択情報","選択数と図面のリビジョン",new WorkshopCommand("selection")),
            new("undo","元に戻す","直前の操作を取り消し（手動操作も対象）",new WorkshopCommand("undo"))]) ]));
    }
    public void OnShutdown() { }
}
public sealed class WorkshopCommand(string action) : IAddinCommand
{
    public async Task<CommandResult> ExecuteAsync(ICadApi api,CancellationToken cancellationToken)
    {
        if(action=="selection")
        {
            var selected=await api.GetSelectionAsync(cancellationToken:cancellationToken);
            return new($"選択 {selected.Total} 個 / revision {selected.Document.Revision}");
        }
        var document=await api.GetDocumentAsync(cancellationToken);
        if(action=="undo")
        {
            await api.UndoAsync(new(document.Id,document.Revision,Guid.NewGuid().ToString("D")),cancellationToken);
            return new("直前の操作を取り消しました。");
        }
        if(api is not ICadJobApi automation)throw new InvalidOperationException("API v1.2 対応の JDesignCAD が必要です。");
        if(action=="curves")
        {
            var request=document.Operation("geometry.create",new CreateObjectsParameters([
                new(Line:new(new(0,0,0),new(50,0,0))),new(Circle:new(new(25,20,0),10,new(0,0,1))) ]));
            await automation.ExecuteOperationAsync(request,cancellationToken);
            return new("線分と円を作成しました。");
        }
        var operation=action=="box"?document.Operation("solid.primitive",new
PrimitiveParameters("box",new(0,0,0),new(20,30,10)))
            :document.Operation("sheet.loft",new
SheetLoftParameters(new(),new(Kind:3,FormingMode:1,Height:80,BottomWidth:100,BottomDepth:80,TopWidth:60,TopDepth:40,Co
rnerRadius:10,TopCornerRadius:6)));
        var capability=(await
automation.GetCapabilitiesAsync(cancellationToken)).Operations.Single(o=>o.Name==operation.Operation);
        if(!capability.Available)throw new InvalidOperationException(capability.UnavailableReason);
        var job=await automation.StartOperationAsync(new(operation,120),cancellationToken);
        try
        {
            while(job.State=="running")
            {
                await Task.Delay(250,cancellationToken);
                job=await automation.GetJobAsync(job.Id,cancellationToken);
            }
            if(job.State!="ready")throw new InvalidOperationException(job.ErrorCode+": "+job.Message);
```

```
            // Refuse to apply a result to a drawing changed during calculation.
            var current=await api.GetDocumentAsync(cancellationToken);
            await
automation.CommitJobAsync(new(current.Id,current.Revision,Guid.NewGuid().ToString("D"),job.Id),cancellationToken);
        }
        catch
        {
            try { await automation.CancelJobAsync(job.Id,CancellationToken.None); } catch { /* CAD may be closing. */ }
            throw;
        }
        return new("作成しました。1 回の［元に戻す］で取り消せます。");
    }
}
```

VB.NET の実装全体

同梱 SDK のコードです。対応するプロジェクトと addin.json を使い、STD／Pro でビルドした DLL を読み込みます。行の折り返しは紙面上の表示です。

```
Imports System.Threading
Imports System.Threading.Tasks
Imports JDesignCAD.Api

Public NotInheritable Class WorkshopAddin
    Implements ICadAddin
    Public Sub OnStartup(context As IAddinContext) Implements ICadAddin.OnStartup
        Dim button = New RibbonButton("line", "線分作成", "50 mm の線分を作成", New LineCommand())
        Dim panel = New RibbonPanel("basic", "VB.NET", New RibbonButton() {button})
        context.RegisterTab(New RibbonTab("workshop-vb", "API 実習 VB", New RibbonPanel() {panel}))
    End Sub
    Public Sub OnShutdown() Implements ICadAddin.OnShutdown
    End Sub
End Class

Public NotInheritable Class LineCommand
    Implements IAddinCommand
    Public Async Function ExecuteAsync(api As ICadApi, cancellationToken As CancellationToken) As Task(Of
CommandResult) Implements IAddinCommand.ExecuteAsync
        Dim doc = Await api.GetDocumentAsync(cancellationToken)
        Dim request = New CreateGeometryRequest(doc.Id, doc.Revision, Guid.NewGuid().ToString("D"),
            New LineDefinition() {New LineDefinition(New Point3(0, 0, 0), New Point3(50, 0, 0))},
            Array.Empty(Of CircleDefinition)())
        Await api.CreateGeometryAsync(request, cancellationToken)
        Return New CommandResult("線分を作成しました。元に戻すで取り消せます。")
    End Function
End Class
```

第 9 部 MCP 連携ガイド

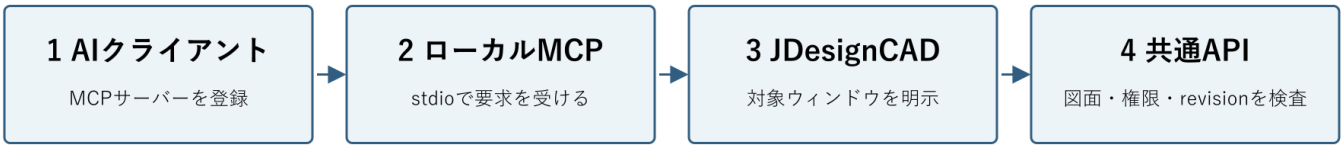


図 130　AI から CAD へ要求が届く経路。

ファイル入出力：保存・書き出し、5 メソッドと MCP、形式・完了・エラーの詳細

Free 版とカスタマイズの利用条件

Free 版は配布時の「API サンプル」10 コマンドだけを利用できます。独自 DLL と変更したサンプルの実行には、有効な STD または Pro ライセンスが必要です。Free 版ではサンプルの名前・ID を使っても外部 DLL のコードは実行せず、本体に同梱されたサンプルを実行します。変更が反映されない場合は［読込状況］とプランを確認してください。C#／VB.NET の実習サンプルを自作 DLL として読み込む場合も STD／Pro が必要です。

MCP は Free 版でも使用できます。通常の作図・編集 API は利用でき、板金・解析の API はライセンス判定で拒否されます。独自 DLL の読込み制限と MCP 接続の可否は別です。

接続設定

本体フォルダーの Customization¥Mcp¥JDesignCAD.Mcp.exe が stdio 方式の MCP サーバーです。MCP クライアントから起動します。TCP の待受ポートは使用せず、同じ Windows ユーザーの本体へ名前付きパイプで接続します。

```
{
  "mcpServers": {
    "jdesigncad": {
      "command": "C:¥¥Program Files¥¥JDesignCADV1¥¥Customization¥¥Mcp¥¥JDesignCAD.Mcp.exe",
      "args": []
    }
  }
}
```

command を実際の配置先へ変更します。同梱の mcp-config.json は配布先に合わせて生成されています。登録方法や設定ファイル名は使用する MCP クライアントに従ってください。

最初の確認

• JDesignCAD を起動し、空の図面をモデル空間で開きます。

• cad_list_sessions で対象ウィンドウの sessionId を選択します。

• cad_get_document と cad_get_capabilities で図面 ID・revision・利用可能な操作を確認します。

• cad_get_operation_schema で操作名の入力スキーマを取得します。

• cad_execute_operation で軽い操作、cad_start_operation で計算ジョブを実行します。

同一部品を分割作成した場合の合体

一つの実物部品を複数のソリッドに分けて作成した場合は、その部品の加工と、分割形状を参照する後続工程が済んだ時点で、作成途中にブーリアン合体します。アセンブリ全体が完成するまで分割したままにせず、

solid.boolean に kind: "union" と対象の entityIds を渡します。例えば、一体のアームを構成する筒・腕・取付部や、同じボルトの軸と頭部が対象です。

• 実物で一つの部品かを確認し、その部品を構成する接続したソリッドだけを選びます。

• cad_start_operation で合体を計算し、cad_get_job で成功を確認します。ready の結果を cad_commit_job で確定します。

• 結果が 1 ソリッドになったことと形状を確認します。元の図形 ID を使い続けず、確定後の ID・revision を使って次の加工へ進みます。

同じ形状を複数配置した別の部品、ナット、ワッシャー、ブッシュ、可動部品などは相互に合体しません。ユーザーが分離を指定したものはそのまま残します。形状間に隙間がある場合、許可なく補形状を追加したり寸法を変えたりしてつなげてはいけません。

合体が失敗した場合は元の形状を保持し、未完了であることと原因・エラーを報告します。ブロック化・グループ化や単なる重ね合わせは、ブーリアン合体の代わりにはなりません。この方針は MCP の接続時の案内と、機能一覧・計算開始ツールの説明でも AI へ伝えます。更新済み MCP サーバーへの再接続後に新しい案内が有効になります。

線分を 1 本作る入力例

ツールは cad_execute_operation です。角括弧の文字列を実際の ID・数値・新しい UUID に置き換えてください。

```
{
  "sessionId": "[対象セッション ID]",
  "request": {
    "documentId": "[図面 ID]", "expectedRevision": 0,
    "operationId": "[新しい UUID]", "operation": "geometry.create",
    "parameters": {
      "objects": [{"line": {
        "start": {"x":0,"y":0,"z":0},
        "end": {"x":50,"y":0,"z":0}
      }}]
    },
    "selectResult": true
  }
}
```

計算ジョブ

cad_start_operation に{request: OperationRequest, timeoutSeconds: 120}を渡します。cad_get_job を適度な間隔で呼び、running が終わるのを待ちます。ready になったら図面の revision が計算開始時と一致することを確認し、cad_commit_job へ新しい operationId を付けて渡します。ジョブ開始の operationId を commit に使い回してはいけません。

計測・解析は completed になり、cad_get_job_data で結果をページ取得します。cad_cancel_job は計算中の中止または ready 結果の破棄です。元の図面は変わりません。接続が切れても計算は自動中止されないため、同じ開始リクエストを再送してジョブ ID を確認できます。

ツールの対応

用途	ツール
接続・情報	cad_list_sessions / cad_get_document / cad_get_capabilities / cad_get_operation_schema
選択・図形取得	cad_get_selection / cad_get_entities / cad_get_entity_details / cad_get_topology / cad_get_attributes / cad_get_layers / cad_query
即時編集	cad_create_geometry / cad_create_curves / cad_set_selection / cad_translate_entities / cad_delete_entities / cad_execute_operation
計算	cad_start_operation / cad_get_job / cad_cancel_job / cad_commit_job / cad_get_job_data
履歴	cad_undo / cad_redo
画面操作	cad_get_interactive_commands / cad_launch_interactive_command

自動処理と画面操作の区別

cad_launch_interactive_command は、ユーザーが CAD 画面で続けて操作するための入口です。queued だけで「保存した」「穴を開けた」「解析が成功した」と判断してはいけません。自動処理には cad_get_capabilities に掲載された構造化操作を優先します。Free では板金・解析の呼出しを拒否します。

ライセンス制限の応答

cad_get_capabilities の license.currentPlan に現在のプラン、各操作の available に利用可否が入ります。制限対象は unavailableCode=license_required、licenseRestriction に feature・operation・currentPlan・requiredPlans・retryable=false を返します。cad_get_operation_schema と対話コマンド一覧でも同じ条件を確認できます。実行時にも判定し、計算ジョブは確定時にも再判定します。

```
{
  "code": "license_required",
  "message": "この API には有効な STD または Pro ライセンスが必要です。",
  "licenseRestriction": {
    "feature": "sheetmetal", "operation": "sheet.loft",
    "currentPlan": "Free", "requiredPlans": ["STD", "Pro"],
    "retryable": false
  }
}
```

上記は応答形式の例です。MCP では isError=true の本文に JSON で返します。AI は制限理由と必要なプランを利用者へ説明し、同じ要求を繰り返したり別 API で制限を回避したりせず、available=true の操作を選び直します。図面は変更されません。通信切断の再試行と、ライセンス制限の拒否を区別してください。

上限と復旧

ローカル通信は 1 メッセージ 1 MiB まで。図形一覧は最大 1000 件、面・辺とジョブ結果は最大 256 件ずつ取得します。巨大な解析入力や 1 フレームが大きい非定常熱結果は MCP 転送の上限に達する場合があります。新しい名前の操作や任意の内部メソッドは実行できません。

通信断後は同じリクエスト・同じ operationId で再試行します。新しい UUID で盲目的に再実行すると、二重作図・二重加工になります。cad_undo は手動編集も含む直前の操作を取り消します。MCP 操作だけに限定されません。

保存・書き出し

cad_get_file_formats → cad_save_document または cad_export_document → cad_get_file_operation。中止は cad_cancel_file_operation。completed で初めて完了です。failed は error を確認し、ファイル処理に cad_commit_job は使いません。Free で許可される形式は JDC／JSON／STL です。

MCP で図面を確認する

view.capture を cad_execute_operation から実行すると、現在のモデルまたは指定したレイアウトを PNG へ描画します。画面のスクリーンショットではなく、CAD の描画経路で投影図・紙面の図形・寸法を出力します。図形、選択、revision は変更しません。レイアウトを表示している状態でも利用できます。

```
{
  "operation": "view.capture",
  "parameters": {
    "filePath": "D:/Drawings/preview.png",
    "layoutName": "板金 1_展開図",
    "width": 2100,
    "height": 1485,
    "overwrite": false
  }
}
```

実際の要求には最新の documentId・expectedRevision・新しい operationId も指定します。layoutName を省略するとモデルを出力します。PNG は CAD を実行している PC に保存されます。確認するクライアントで画像を開いてください。出力先は存在するフォルダーの絶対パス、画像サイズは幅 320〜4096・高さ 240〜4096 です。既存ファイルを置き換える場合だけ overwrite を true にします。

cad_query の kind=layouts でレイアウト一覧、kind=sheetDefaults で現在の板金設定を取得できます。layout.create はモデルの投影図、紙面の図形、実寸値の寸法を一括作成し、1 回の Undo で戻せます。layout.activate でレイアウトを切り替えます。曲げフランジの一部区間は sheet.flange の startDistance・endDistance で指定します。入力詳細は各操作のスキーマを取得してください。

注釈と寸法の作図ルール

R、穴数、長穴、面取りなどの注記は、原則として文字付き引き出し線を使用します。独立した文字と線に分けず、文字・矢印・折れ点を一体の編集可能な注釈にします。cad_query の kind=layoutAnnotations で紙面の要素番号を取得し、同じ revision で layout.annotations を呼び出すと、指定注釈の置き換えや追加ができます。Text は本文、Anchor は矢印先端、Landing は文字を載せる水平線の起点、Elbows は途中の折れ点です。寸法・モデル・投影図はそのまま保持します。

円や円弧の位置・ピッチを表す寸法は、その意図に応じて正確な中心を基準にします。投影した数値が同じでも、中心の代わりに接点を選ばないようにします。全長・接線・端面寸法は元図面の指定に従います。ユーザーが維持を指示した寸法は変更しません。確認には view.capture を使用します。

図面を開く・モデルとレイアウトを切り替える

cad_get_capabilities と cad_get_operation_schema で操作を確認し、既存の cad_execute_operation から実行できます。MCP 接続の追加設定は不要です。

• cad_query の kind="views"：モデル／レイアウトの名前、種類、現在表示中か、用紙寸法を取得します。

• view.activate：{"kind":"model"}でモデルに戻り、{"kind":"layout","name":"用紙名"}でレイアウトを表示します。元の倍率を保ちます。全体表示が必要な場合だけ zoomToFit=true を指定します。図形と Undo 履歴は変更しません。

• document.open：{"filePath":"D:/Drawings/drawing.jdc"}で既存の JDC／JDC JSON を同じウィンドウで開きます。読み込みはバックグラウンドで行い、成功後に切り替えます。これは Undo で戻せる編集ではありません。

通常は最新の documentId・expectedRevision と新しい operationId を指定します。ホーム画面で図面を開いていない場合だけ、documentId を空文字列、expectedRevision を 0 にします。未保存の変更がある場合は unsaved_changes で拒否されます。先に cad_save_document を実行し、completed を確認してください。読み込み中に編集・別図面への切り替えがあった場合も、その図面を保護して切り替えを中止します。通信タイムアウト後は同じ要求・同じ operationId で再試行します。重複して開き直しません。成功した応答の document を取得し直し、以前の図形 ID を再利用しないでください。

面へ文字・画像を貼り付ける

注釈ボタン（引き出し線などのプルダウン）で「面にラベル貼付け」を選び、モデルの面をクリックします。文字または画像を指定し、幅・高さ・回転・中心位置をプレビューで調整して確定します。「対象ラベル」から貼付け済みラベルを選び、「操作」を編集または削除に切り替え、右上の［✓］で確定します。面選択や設定変更だけでは確定しません。フォントは現在の図面の文字フォントを初期表示し、一覧から変更できます。削除しても立体は残り、Undo でラベルを戻せます。コマンド ID は MenuId.Constr.SurfaceLabel です。

平面、円筒、球面、自由曲面に対応します。同一曲面の連続した面や円筒の継ぎ目をまたいで表示し、実際の面端と穴では切り抜きます。幅・高さは中心位置の接平面を基準とするため、球面・自由曲面では端にゆがみが出ることがあります。特異点の中心や一周以上を覆う指定は使用できません。

元の立体形状・体積は変更しません。ラベル画像は JDC へ埋め込まれ、元画像ファイルがなくても表示でき、移動・コピーに追従します。印刷と PNG 出力に含まれます。STEP／STL には文字や画像の見た目を出力しません。

MCP では face.label を cad_execute_operation で実行します。最新のトポロジーから取得した face（entityId・revision・kind="face"・index）と WCS の center、mm 単位の width・height、度単位の rotationDegrees、text または絶対パスの imagePath を渡します。文字は fontName と 6 桁 RGB の color も指定できます。

cad_query の kind="surfaceLabels" に対象の entityIds（1〜100 件）を渡すと面参照、ラベル ID、配置を取得できます。既定の mode="add" は追加、replace／remove は labelId を指定して編集／削除します。接続面への表示も 1 回の Undo で戻せます。実行前に cad_get_operation_schema で入力形式を確認してください。

MCP ツールリファレンス

31 ツールを収録します。JSON は camelCase です。sessionId で対象の CAD を明示し、変更前には図面の revision と操作の定義を確認してください。

cad_cancel_file_operation

確定前のファイル操作を中止します。中止後の最終状態を確認してください。

入力	型	必須／既定値
sessionId	string	必須
operationId	string	必須

対応メソッド：CancelFileOperationAsync

cad_cancel_job

running の計算を中止し、ready の未確定結果を破棄します。既に確定した図面の取消しには UndoAsync を使います。

入力	型	必須／既定値
sessionId	string	必須
jobId	string	必須

対応メソッド：CancelJobAsync

cad_commit_job

ready の結果を 1 回の Undo で反映します。開始後に図面が変わっていれば拒否。開始とは別の UUID を使い、ライセンスも再判定します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：CommitJobAsync

cad_create_curves

合計 1〜1000 曲線、ポリライン点は合計 4096 点まで。閉輪郭の終点に始点を重複させません。円弧は右手系の符号付き角度で、0 度と ±360 度は不可です。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：CreateCurvesAsync

cad_create_geometry

Lines と Circles に定義を渡します。未使用の配列は空配列とし、合計 1〜1000 図形を 1 回の Undo で作成します。円の Normal はゼロでない法線です。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：CreateGeometryAsync

cad_delete_entities

対象を情報取得で確認してから、明示した 1〜1000 ID を削除します。ロックされたレイヤーや失効した ID は拒否します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：DeleteEntitiesAsync

cad_execute_operation

Execution=immediate の操作を 1 回の Undo で反映します。worker 操作は StartOperationAsync へ渡します。未知のフィールドは拒否します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：ExecuteOperationAsync

cad_export_document

元の図面を維持してコピーを書き出します。状態照会で completed を確認します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：ExportDocumentAsync

cad_get_attributes

1〜100 ID の色、線種、線幅、レイヤーロックを取得します。色は byLayer／byBlock／foreground／rgb を区別します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetAttributesAsync

cad_get_capabilities

操作名、入力型、実行方式、利用可否、制限理由、現在のプランを取得します。操作名を推測せず、Available を確認してください。

入力	型	必須／既定値
sessionId	string	必須

対応メソッド：GetCapabilitiesAsync

cad_get_document

図面 ID、リビジョン、図形数、Undo／Redo 可否を取得します。編集要求を作る直前に呼び出します。

入力	型	必須／既定値
sessionId	string	必須

対応メソッド：GetDocumentAsync

cad_get_entities

モデル直下の図形を作成順のページで取得します。同じ一覧を読み切る間はリビジョンを固定し、変更があれば最初から取得します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetEntitiesAsync

cad_get_entity_details

線分・円・円弧・直線ポリラインの正確な定義を取得します。1〜100 ID。GeometryStatus が unsupported／too_large なら定義の取得はできていません。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetEntityDetailsAsync

cad_get_file_formats

対応する保存・書き出し形式と利用条件を取得します。

入力	型	必須／既定値
sessionId	string	必須

対応メソッド：GetFileFormatsAsync

cad_get_file_operation

ファイル操作の状態、結果、警告、エラーを取得します。

入力	型	必須／既定値
sessionId	string	必須
operationId	string	必須

対応メソッド：GetFileOperationAsync

cad_get_interactive_commands

登録された 266 個のコマンド ID と利用条件を取得します。これらは画面操作の入口で、自動作図のパラメーターAPI ではありません。

入力	型	必須／既定値
sessionId	string	必須

対応メソッド：GetInteractiveCommandsAsync

cad_get_job

running、ready、completed、failed、cancelled、committed を区別します。ready はまだ図面へ反映されていません。

入力	型	必須／既定値
sessionId	string	必須
jobId	string	必須

対応メソッド：GetJobAsync

cad_get_job_data

completed のデータジョブから summary や結果配列をページ取得します。1〜256 件。データジョブには CommitJobAsync を呼びません。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetJobDataAsync

cad_get_layers

図面内のレイヤー定義と、使用できる線種・線幅名を取得します。属性変更時に存在する名前を選ぶために使います。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetLayersAsync

cad_get_operation_schema

指定した操作の JSON Schema を取得します。必須項目、型、null 許可、入れ子を確認します。プロパティ名は camelCase です。

入力	型	必須／既定値
sessionId	string	必須
operation	string	必須

対応メソッド：GetOperationSchemaAsync

cad_get_selection

現在の選択をページ単位で取得します。ID・型・レイヤー・境界ボックスを返します。offset は 0 始まり、limit は 1〜1000 です。

入力	型	必須／既定値
sessionId	string	必須
offset	integer	0
limit	integer	256

対応メソッド：GetSelectionAsync

cad_get_topology

Kind は face／edge／vertex。ソリッド・シェル・面を対象に 1〜256 件ずつ取得します。SubobjectRef はリビジョン変更で失効します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：GetTopologyAsync

cad_launch_interactive_command

ユーザーが続けて操作するコマンドを起動します。queued は起動予約の受付であり、作図・保存・解析の完了ではありません。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：LaunchInteractiveCommandAsync

cad_list_sessions

同じ Windows ユーザーの起動中 CAD を列挙します。対象セッションを明示して選んでください。

対応メソッド：LocalProtocol.FindSessions

cad_query

Kind=geometry は楕円・スプライン・文字などの詳細、sheet は板金情報、blocks は定義とモデル直下の挿入数、styles はスタイル名です。geometry／sheet は 1〜100 ID、その他は空配列です。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：QueryAsync

cad_redo

CanRedo を確認して実行します。戻り値の Document が更新後の状態です。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：RedoAsync

cad_save_document

現在の図面を保存します。状態照会で completed を確認します。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：SaveDocumentAsync

cad_set_selection

明示した ID だけを選択します。空配列で解除。ハイライトだけを変更し、Undo 履歴を増やしません。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：SetSelectionAsync

cad_start_operation

切り離した形状を別プロセスで計算します。この時点では図面を変更しません。running／ready の保持は最大 4 件、同時実行は 1 件、時間制限は 1〜1800 秒です。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：StartOperationAsync

cad_translate_entities

1〜1000 個の図形を WCS の Displacement だけ移動します。Copy=true で元を保持。属性を引き継ぎ、全体を 1 回の Undo とします。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：TranslateEntitiesAsync

cad_undo

手動操作も含む図面の直前の履歴が対象です。API が作った図形だけを取り消す機能ではありません。

入力	型	必須／既定値
sessionId	string	必須
request	object	必須

対応メソッド：UndoAsync

第 10 部 API メソッドと操作のリファレンス

コード例を動かす準備　511

設定と接続　512

作画：線・円・閉輪郭　518

選択と図形の調査　521

入力と対話コマンド　529

編集・複写・元に戻す　534

3D とバックグラウンド処理　539

板金　544

計測と解析　545

レイアウト・注記・画面確認　548

読み込み・保存・書き出し　549

コマンド登録とアドイン　551

メソッド名索引　554

設定、作画、選択、入力、編集、3D、板金、解析、レイアウト、保存の順で説明します。メソッドの引数表と具体例を同じ分類にまとめています。

コード例を動かす準備

同梱 ApiExamples の Samples.cs は JDesignCAD.Api.dll を参照してコンパイルできます。使用する名前空間は System、System.Linq、System.Threading、System.Threading.Tasks、System.Collections.Generic、System.Text.Json、JDesignCAD.Api です。.NET 8 の ImplicitUsings を有効にします。外部接続は Connect(processId)、アドイン内は IAddinContext.Api を ICadFileApi へキャストして cad へ代入します。CancellationToken は CancellationToken.None または自分の CancellationTokenSource から渡します。

例は練習用の新規図面で実行してください。OperationId は新しい UUID です。通信断の再送だけは同じ要求と同じ UUID を使います。競合を隠すために revision だけ更新して再送しないでください。エラーは CadApiException.Code で判別し、利用不可・ライセンス制限・形状不成立は理由を確認します。

ApiExamples フォルダーで dotnet build を実行します。JDesignCAD で練習用図面を開き、dotnet run -- -- list で対象の ProcessId を調べます。例：dotnet run -- 1234 drawing は指定した CAD に線と円を作成します（1234 は実際の ProcessId へ置換）。inspect は選択状態の調査、rectangle は閉輪郭、box は直方体、sheet は板金ベースを作ります。実行の入口は Program.cs、各処理は Samples.cs にあります。

設定と接続

Connect には FindSessions で表示した ProcessId を渡します。複数起動中に最初のセッションを無条件で選ばないでください。長さは表示単位に関係なく WCS の mm、角度は度です。SettingsAsync は Example レイヤを作成・更新します。戻り値 Layers で反映を確認できます。

コード例：connect

```
public static ICadFileApi Connect(int processId)
 {
     SessionInfo session = LocalProtocol.FindSessions()
         .Single(s => s.ProcessId == processId);
     return new LocalCadClient(session);
 }
```

コード例：settings

```
public static async Task<LayerPage> SettingsAsync(
     ICadAutomationApi cad, CancellationToken ct = default)
 {
     CapabilityInfo caps = await cad.GetCapabilitiesAsync(ct);
     if (!caps.Operations.Any(o => o.Name == "layers.set" && o.Available))
         throw new InvalidOperationException("layers.set is unavailable.");
     OperationSchema schema = await cad.GetOperationSchemaAsync("layers.set", ct);
     Console.WriteLine(schema.ParametersSchema.GetRawText());
     DocumentInfo doc = await cad.GetDocumentAsync(ct);
     EditResult edit = await cad.ExecuteOperationAsync(doc.Operation("layers.set",
         new SetLayerParameters("Example", new CadColor("rgb", "0080C0"))), ct);
     return await cad.GetLayersAsync(
         new DocumentRequest(edit.Document.Id, edit.Document.Revision), ct);
 }
```

ICadApi.GetDocumentAsync

図面の状態を取得 — 図面 ID、リビジョン、図形数、Undo／Redo 可否を取得します。編集要求を作る直前に呼び出します。

宣言

```
Task<DocumentInfo> GetDocumentAsync(

    CancellationToken cancellationToken = default

);
```

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<DocumentInfo>。Task は await／Await で待機します。

プロパティ	型
Id	string

プロパティ	型
FileName	string
Revision	long
ObjectCount	int
CanUndo	bool
CanRedo	bool
LengthUnit	string
CoordinateSystem	string

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.GetCapabilitiesAsync

操作と利用条件を取得 — 操作名、入力型、実行方式、利用可否、制限理由、現在のプランを取得します。操作名を推測せず、Available を確認してください。

宣言

```
Task<CapabilityInfo> GetCapabilitiesAsync(
    CancellationToken cancellationToken = default
);
```

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<CapabilityInfo>。Task は await／Await で待機します。

プロパティ	型
Major	int

プロパティ	型
Minor	int
CoordinateSystem	string
LengthUnit	string
Operations	OperationInfo[]
Limitations	string[]
License	LicenseCapabilities

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.GetOperationSchemaAsync

操作の入力書式を取得 — 指定した操作の JSON Schema を取得します。必須項目、型、null 許可、入れ子を確認します。プロパティ名は camelCase です。

宣言

```
Task<OperationSchema> GetOperationSchemaAsync(
    string operation,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
operation	string	GetCapabilitiesAsync にある登録済みの操作名。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<OperationSchema>。Task は await／Await で待機します。

プロパティ	型
Operation	OperationInfo
ParametersSchema	Text.Json.JsonElement

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.GetLayersAsync

レイヤーと線の設定を取得 — 図面内のレイヤー定義と、使用できる線種・線幅名を取得します。属性変更時に存在する名前を選ぶために使います。

宣言

```
Task<LayerPage> GetLayersAsync(
    DocumentRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	DocumentRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<LayerPage>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Layers	LayerInfo[]
LinePatterns	string[]

プロパティ	型
LineWidths	string[]

要求型

```
public sealed record DocumentRequest(string DocumentId, long ExpectedRevision);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.ExecuteOperationAsync

即時操作を実行 — Execution=immediate の操作を 1 回の Undo で反映します。worker 操作は StartOperationAsync へ渡します。未知のフィールドは拒否します。

宣言

```
Task<EditResult> ExecuteOperationAsync(

    OperationRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	OperationRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record OperationRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string Operation, JsonElement Parameters, bool SelectResult = true);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

作画：線・円・閉輪郭

DrawingAsync は長さ 60 の線と半径 10 の円を 1 回の Undo 単位で作ります。RectangleAsync は 80×50 の閉じたポリラインを作ります。Closed=true では最初の頂点を末尾に重ねません。戻り値 Entities の Id を後続編集に渡します。

コード例：drawing

```
public static async Task<EditResult> DrawingAsync(
    ICadApi cad, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    var request = new CreateGeometryRequest(doc.Id, doc.Revision,
        Guid.NewGuid().ToString("D"),
        new[] { new LineDefinition(new(0, 0, 0), new(60, 0, 0)) },
        new[] { new CircleDefinition(new(30, 20, 0), 10, new(0, 0, 1)) });
    EditResult result = await cad.CreateGeometryAsync(request, ct);
    Console.WriteLine($"Created: {result.Entities.Length}; revision: {result.Document.Revision}");
    return result;
}
```

コード例：curves

```
public static async Task<EditResult> RectangleAsync(
    ICadModelApi cad, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    var rectangle = new PolylineDefinition(new Point3[] {
        new(0, 0, 0), new(80, 0, 0), new(80, 50, 0), new(0, 50, 0)
    }, Closed: true);
    return await cad.CreateCurvesAsync(new CreateCurvesRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"),
        new[] { rectangle }, Array.Empty<ArcDefinition>()), ct);
}
```

ICadApi.CreateGeometryAsync

線分と円を作成 — Lines と Circles に定義を渡します。未使用の配列は空配列とし、合計 1〜1000 図形を 1 回の Undo で作成します。円の Normal はゼロでない法線です。

宣言

```
Task<EditResult> CreateGeometryAsync(

    CreateGeometryRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	CreateGeometryRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record CreateGeometryRequest(string DocumentId, long ExpectedRevision, string OperationId,
    LineDefinition[] Lines, CircleDefinition[] Circles, bool SelectCreated = true);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadModelApi.CreateCurvesAsync

ポリラインと円弧を作成 — 合計 1〜1000 曲線、ポリライン点は合計 4096 点まで。閉輪郭の終点に始点を重複させません。円弧は右手系の符号付き角度で、0 度と ±360 度は不可です。

宣言

```
Task<EditResult> CreateCurvesAsync(
    CreateCurvesRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	CreateCurvesRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record CreateCurvesRequest(string DocumentId, long ExpectedRevision, string OperationId,
    PolylineDefinition[] Polylines, ArcDefinition[] Arcs, bool SelectCreated = true);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

選択と図形の調査

選択取得の戻り値は全件とは限りません。Total と Offset を確認し、大量の場合はページを分けて取得します。調査中は同じ revision を保ちます。面・エッジの SubobjectRef は形状編集後に取り直し、固定した Index を別形状へ使わないでください。

コード例：selection

```
public static async Task<SelectionInfo> SelectCreatedAsync(
    ICadModelApi cad, CancellationToken ct = default)
{
    EditResult created = await DrawingAsync(cad, ct);
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    await cad.SetSelectionAsync(new EntityEditRequest(doc.Id, doc.Revision,
        Guid.NewGuid().ToString("D"), created.Entities.Select(e => e.Id).ToArray()), ct);
    SelectionInfo selection = await cad.GetSelectionAsync(0, 256, ct);
    Console.WriteLine($"Selected {selection.Entities.Length} of {selection.Total}");
    return selection;
}
```

コード例：inspect

```
public static async Task InspectAsync(
    ICadAutomationApi cad, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    int offset = 0;
    while (true)
    {
        EntityPage page = await cad.GetEntitiesAsync(
            new EntityQuery(doc.Id, doc.Revision, offset, 100), ct);
        if (page.Entities.Length == 0) break;
        var request = new EntityDetailsRequest(doc.Id, doc.Revision,
            page.Entities.Select(e => e.Id).ToArray());
        EntityDetailsResult details = await cad.GetEntityDetailsAsync(request, ct);
        AttributePage attributes = await cad.GetAttributesAsync(request, ct);
        foreach (EntityDetail item in details.Entities)
            Console.WriteLine($"{item.Entity.Id}: {item.GeometryStatus}");
        foreach (AttributeInfo item in attributes.Items)
            Console.WriteLine($"Layer: {item.Entity.Layer}; locked: {item.LayerLocked}");
        offset += page.Entities.Length;
        if (offset >= page.Total) break;
    }
}
```

コード例：topology

```
public static async Task<TopologyItem[]> EdgesAsync(
    ICadAutomationApi cad, string entityId, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    var edges = new List<TopologyItem>();
    do
    {
        TopologyPage page = await cad.GetTopologyAsync(new TopologyRequest(
            doc.Id, doc.Revision, entityId, "edge", edges.Count, 128), ct);
        edges.AddRange(page.Items);
        if (edges.Count >= page.Total || page.Items.Length == 0) break;
    } while (true);
    return edges.ToArray();
}
```

ICadApi.GetSelectionAsync

選択図形を取得 — 現在の選択をページ単位で取得します。ID・型・レイヤー・境界ボックスを返します。offset は 0 始まり、limit は 1〜1000 です。

宣言

```
Task<SelectionInfo> GetSelectionAsync(
    int offset = 0,
    int limit = 256,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
offset	int	先頭から飛ばす件数。0 始まり。
limit	int	1 ページの最大件数。このメソッドの上限内で指定。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<SelectionInfo>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Total	int
Offset	int

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadModelApi.GetEntitiesAsync

図面の図形一覧を取得 — モデル直下の図形を作成順のページで取得します。同じ一覧を読み切る間はリビジョンを固定し、変更があれば最初から取得します。

宣言

```
Task<EntityPage> GetEntitiesAsync(
    EntityQuery request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	EntityQuery	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EntityPage>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Total	int
Offset	int

要求型

```
public sealed record EntityQuery(string DocumentId, long ExpectedRevision, int Offset = 0, int Limit = 256);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadModelApi.GetEntityDetailsAsync

線や円の定義を取得 — 線分・円・円弧・直線ポリラインの正確な定義を取得します。1〜100 ID。GeometryStatus が unsupported／too_large なら定義の取得はできていません。

宣言

```
Task<EntityDetailsResult> GetEntityDetailsAsync(
    EntityDetailsRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	EntityDetailsRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EntityDetailsResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityDetail[]

要求型

```
public sealed record EntityDetailsRequest(string DocumentId, long ExpectedRevision, string[] EntityIds);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadModelApi.SetSelectionAsync

選択を置き換える — 明示した ID だけを選択します。空配列で解除。ハイライトだけを変更し、Undo 履歴を増やしません。

宣言

```
Task<EditResult> SetSelectionAsync(
    EntityEditRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	EntityEditRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record EntityEditRequest(string DocumentId, long ExpectedRevision, string OperationId, string[] EntityIds);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.GetAttributesAsync

図形の属性を取得 — 1〜100 ID の色、線種、線幅、レイヤーロックを取得します。色は byLayer／byBlock／foreground／rgb を区別します。

宣言

```
Task<AttributePage> GetAttributesAsync(
    EntityDetailsRequest request,
```

```
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	EntityDetailsRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<AttributePage>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Items	AttributeInfo[]

要求型

```
public sealed record EntityDetailsRequest(string DocumentId, long ExpectedRevision, string[] EntityIds);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadAutomationApi.GetTopologyAsync

面や辺の参照を取得 — Kind は face／edge／vertex。ソリッド・シェル・面を対象に 1〜256 件ずつ取得します。SubobjectRef はリビジョン変更で失効します。

宣言

```
Task<TopologyPage> GetTopologyAsync(
    TopologyRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	TopologyRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<TopologyPage>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Items	TopologyItem[]
Total	int
Offset	int

要求型

```
public sealed record TopologyRequest(string DocumentId, long ExpectedRevision, string EntityId,
    string Kind = "face", int Offset = 0, int Limit = 128);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadWorkspaceApi.QueryAsync

拡張情報を取得 — Kind=geometry は楕円・スプライン・文字などの詳細、sheet は板金情報、blocks は定義とモデル直下の挿入数、styles はスタイル名です。geometry／sheet は 1〜100 ID、その他は空配列です。

宣言

```
Task<QueryResult> QueryAsync(
    QueryRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	QueryRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<QueryResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Data	Text.Json.JsonElement

要求型

```
public sealed record QueryRequest(string DocumentId,long ExpectedRevision,string Kind,string[] EntityIds);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

入力と対話コマンド

独自コマンドで入力を受け取る

ICadInputApi は、標準作画コマンドを起動せずにユーザーの入力を返します。GetPointAsync 等を await／Await し、Accepted を確認してから返された Point や Entities を使って図形を作成・編集します。UI スレッドで.Wait()／.Result は使用しません。入力結果の Document を次の要求へ引き継ぎます。

Esc は cancelled、許可した空入力は none です。CancellationToken の中止は OperationCanceledException になります。カーソル位置は未確定です。入力中に図面・ビュー・形状が変わると中止されます。仮表示は保存・Undo の対象になりません。

座標は WCS の mm、角度は度です。角度と長方形は現在の作図平面を使います。窓選択は左→右、交差選択は右→左です。種類とレイヤはリスト内 OR、リスト間 AND で判定します。面・エッジ参照は形状編集後に取得し直してください。

完全なプロジェクトは ApiExamples/InputCS・InputVB、または Customization/SDK/Samples/Customization.InputCS・Customization.InputVB です。2 点線分、円プレビュー、選択複写の 3 コマンドを収録します。独自 DLL の実行は STD／Pro が対象です。

メソッド	用途と戻り値
GetPointAsync	点。基点を渡すとラバーバンド。現在のスナップを使用。
GetCornerAsync	現在の作図平面上に長方形を仮表示し、対角点を返す。
GetDistanceAsync / GetAngleAsync	基点とマウス位置、または数値入力。Number は mm／度。
GetIntegerAsync / GetDoubleAsync	整数／実数。既定値、最小・最大を指定。
GetStringAsync / GetKeywordsAsync	文字／候補語。Text に結果。キーワードは一意の前方一致可。
GetEntityAsync / GetSelectionAsync	単一／複数選択。Enter で確定。Kinds・Layers で絞り込み。
GetSubentityAsync	face／edge を指定。Subobjects にトポロジー参照。
SelectRegionAsync	all／window／crossing。非表示・ロック対象の条件を確認。
DragAsync / UpdateInputPreviewAsync	カーソル位置から線・円の仮表示。最大 256 図形。
StartInputAsync / GetInputAsync / CancelInputAsync	MCP 等で開始・状態取得・中止を分離。

2 点から線分を作る

以下は IAddinCommand の実装です。C#は System、System.Threading、System.Threading.Tasks、JDesignCAD.Api を using し、VB.NET は同じ名前空間を Imports します。名前空間・登録処理・addin.json を含む完全なファイルは同梱プロジェクトにあります。

C#

```
public sealed class TwoPointLineCommand : IAddinCommand
{
    public async Task<CommandResult> ExecuteAsync(ICadApi api,CancellationToken cancellationToken)
    {
        var input=(ICadInputApi)api;
        var doc=await api.GetDocumentAsync(cancellationToken);

        // 1 点目をユーザーが指示するまで待つ。
        var first=await input.GetPointAsync(doc,"1 点目を指示してください",cancellationToken:cancellationToken);
        if(!first.Accepted)return new CommandResult("");

        // 1 点目からマウス位置へラバーバンドを表示し、2 点目を待つ。
        var second=await input.GetPointAsync(first.Document,"2 点目を指示してください",
            basePoint:first.Point,cancellationToken:cancellationToken);
        if(!second.Accepted)return new CommandResult("");

        var a=first.Point!;var b=second.Point!;
        if(Math.Sqrt(Math.Pow(b.X-a.X,2)+Math.Pow(b.Y-a.Y,2)+Math.Pow(b.Z-a.Z,2))<=1e-6)
            return new CommandResult("異なる 2 点を指示してください。");

        // ここからが自作の作図処理。標準の線分コマンドは呼ばない。
        await api.CreateGeometryAsync(new CreateGeometryRequest(
            second.Document.Id,second.Document.Revision,Guid.NewGuid().ToString("D"),
            new[]{new LineDefinition(a,b)},Array.Empty<CircleDefinition>()),cancellationToken);
        return new CommandResult("");
    }
}
```

VB.NET

```
Public NotInheritable Class TwoPointLineCommand
        Implements IAddinCommand
        Public Async Function ExecuteAsync(api As ICadApi, cancellationToken As CancellationToken) As Task(Of
CommandResult) Implements IAddinCommand.ExecuteAsync
            Dim input = DirectCast(api, ICadInputApi)
            Dim doc = Await api.GetDocumentAsync(cancellationToken)

            ' 1 点目をユーザーが指示するまで待つ。
            Dim first = Await input.GetPointAsync(doc, "1 点目を指示してください", cancellationToken:=cancellationToken)
            If Not first.Accepted Then Return New CommandResult("")

            ' 1 点目からマウス位置へラバーバンドを表示し、2 点目を待つ。
            Dim second = Await input.GetPointAsync(first.Document, "2 点目を指示してください", basePoint:=first.Point,
cancellationToken:=cancellationToken)
            If Not second.Accepted Then Return New CommandResult("")
            Dim a = first.Point
            Dim b = second.Point
            If Math.Sqrt((b.X - a.X) ^ 2 + (b.Y - a.Y) ^ 2 + (b.Z - a.Z) ^ 2) <= 0.000001 Then Return New
CommandResult("異なる 2 点を指示してください。")

            ' ここからが自作の作図処理。標準の線分コマンドは呼ばない。
            Await api.CreateGeometryAsync(New CreateGeometryRequest(
                second.Document.Id, second.Document.Revision, Guid.NewGuid().ToString("D"),
                New LineDefinition() {New LineDefinition(a, b)}, Array.Empty(Of CircleDefinition)()), cancellationToken)
            Return New CommandResult("")
        End Function
    End Class
```

入れ子ブロック内の図形選択、フェンス／多角形選択、点入力中のキーワード分岐、任意の 3D 形状を使う Jig はこの API の対象外です。AutoCAD と同じバイナリを使用する API ではありません。

MCP では cad_start_input で開始し、cad_get_input で状態を取得します。cad_cancel_input で中止、cad_update_input_preview で仮表示を更新、cad_select_region で範囲選択します。waiting を入力完了と解釈せず、accepted の結果だけを確定座標として使います。

ICadInputApi は、標準作画コマンドを起動せずにユーザーの入力を返します。GetPointAsync 等を await／Await し、Accepted を確認してから返された Point や Entities を使って図形を作成・編集します。UI スレッドで.Wait()／.Result は使用しません。入力結果の Document を次の要求へ引き継ぎます。

コード例：input

```
public static async Task<InteractiveCommandResult> LaunchInputAsync(
    ICadInteractionApi cad, string commandId,
    CancellationToken ct = default)
{
    InteractiveCommandInfo[] commands = await cad.GetInteractiveCommandsAsync(ct);
    InteractiveCommandInfo command = commands.Single(c => c.Id == commandId);
    if (!command.Available)
        throw new InvalidOperationException(command.UnavailableReason);
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    return await cad.LaunchInteractiveCommandAsync(new InteractiveCommandRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"), command.Id), ct);
}
```

ICadInteractionApi.GetInteractiveCommandsAsync

画面コマンドを取得 — 登録された 266 個のコマンド ID と利用条件を取得します。これらは画面操作の入口で、自動作図のパラメーターAPI ではありません。

宣言

```
Task<InteractiveCommandInfo[]> GetInteractiveCommandsAsync(

    CancellationToken cancellationToken = default

);
```

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<InteractiveCommandInfo[]>。Task は await／Await で待機します。

プロパティ	型
Id	string
Group	string
Available	bool
UnavailableReason	string

プロパティ	型
UnavailableCode	string
LicenseRestriction	LicenseRestriction

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadInteractionApi.LaunchInteractiveCommandAsync

画面コマンドを起動 — ユーザーが続けて操作するコマンドを起動します。queued は起動予約の受付であり、作図・保存・解析の完了ではありません。

宣言

```
Task<InteractiveCommandResult> LaunchInteractiveCommandAsync(
    InteractiveCommandRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	InteractiveCommandRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<InteractiveCommandResult>。Task は await／Await で待機します。

プロパティ	型
CommandId	string
State	string
Message	string

要求型

```
public sealed record InteractiveCommandRequest(string DocumentId,long ExpectedRevision,string OperationId,string CommandId);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

編集・複写・元に戻す

CopySelectionAsync は選択図形を X 方向へ 100 mm 複写し、元図形を残します。削除・Undo・Redo の例は、このコードで新規作成した図形だけを対象にします。他の操作と Undo を混在させず、各戻り値 Document の revision を引き継ぎます。

コード例：edit

```
public static async Task<EditResult> CopySelectionAsync(
    ICadModelApi cad, CancellationToken ct = default)
{
    SelectionInfo selected = await cad.GetSelectionAsync(0, 256, ct);
    if (selected.Total == 0 || selected.Total > selected.Entities.Length)
        throw new InvalidOperationException("Select 1 to 256 complete model objects.");
    DocumentInfo doc = selected.Document;
    return await cad.TranslateEntitiesAsync(new TranslateEntitiesRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"),
        selected.Entities.Select(e => e.Id).ToArray(),
        new Point3(100, 0, 0), Copy: true), ct);
}
```

コード例：history

```
public static async Task DeleteOwnExampleAndUndoAsync(
    ICadModelApi cad, CancellationToken ct = default)
{
    EditResult made = await DrawingAsync(cad, ct);
    DocumentInfo doc = made.Document;
    EditResult deleted = await cad.DeleteEntitiesAsync(new EntityEditRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"),
        made.Entities.Select(e => e.Id).ToArray()), ct);
    doc = deleted.Document;
    EditResult undone = await cad.UndoAsync(new HistoryRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D")), ct);
    doc = undone.Document;
    if (doc.CanRedo)
        await cad.RedoAsync(new HistoryRequest(
            doc.Id, doc.Revision, Guid.NewGuid().ToString("D")), ct);
}
```

ICadModelApi.TranslateEntitiesAsync

移動または複写 — 1〜1000 個の図形を WCS の Displacement だけ移動します。Copy=true で元を保持。属性を引き継ぎ、全体を 1 回の Undo とします。

宣言

```
Task<EditResult> TranslateEntitiesAsync(

    TranslateEntitiesRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	TranslateEntitiesRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record TranslateEntitiesRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string[] EntityIds, Point3 Displacement, bool Copy = false, bool SelectResult = true);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadModelApi.DeleteEntitiesAsync

図形を削除 — 対象を情報取得で確認してから、明示した 1〜1000 ID を削除します。ロックされたレイヤーや失効した ID は拒否します。

宣言

```
Task<EditResult> DeleteEntitiesAsync(
    EntityEditRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	EntityEditRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record EntityEditRequest(string DocumentId, long ExpectedRevision, string OperationId, string[]
EntityIds);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadApi.UndoAsync

直前の操作を取り消す — 手動操作も含む図面の直前の履歴が対象です。API が作った図形だけを取り消す機能ではありません。

宣言

```
Task<EditResult> UndoAsync(

    HistoryRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	HistoryRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record HistoryRequest(string DocumentId, long ExpectedRevision, string OperationId);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadApi.RedoAsync

取り消した操作を戻す — CanRedo を確認して実行します。戻り値の Document が更新後の状態です。

宣言

```
Task<EditResult> RedoAsync(
    HistoryRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	HistoryRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record HistoryRequest(string DocumentId, long ExpectedRevision, string OperationId);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

3D とバックグラウンド処理

RunEditAsync は開始→状態確認→ready→確定までを担当する共通関数です。計算中に図面が変わると確定できません。その場合は選択と参照を取り直して最初から計算します。キャンセル時は専用 CancelJobAsync でワーカーも停止します。BoxAsync の例は 80×50×20 mm です。

コード例：worker

```
public static async Task<EditResult> RunEditAsync<T>(
    ICadJobApi cad, string operation, T parameters,
    CancellationToken ct = default)
 {
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    JobInfo job = await cad.StartOperationAsync(new StartOperationRequest(
        doc.Operation(operation, parameters), TimeoutSeconds: 120), ct);
    try
    {
        while (job.State == "running")
        {
            await Task.Delay(200, ct);
            job = await cad.GetJobAsync(job.Id, ct);
        }
        if (job.State != "ready")
            throw new InvalidOperationException($"{job.State}: {job.ErrorCode}: {job.Message}");
        return await cad.CommitJobAsync(new CommitJobRequest(
            doc.Id, doc.Revision, Guid.NewGuid().ToString("D"), job.Id), ct);
    }
    catch (OperationCanceledException)
    {
        await cad.CancelJobAsync(job.Id, CancellationToken.None);
        throw;
    }
 }
```

コード例：solid

```
public static Task<EditResult> BoxAsync(
    ICadJobApi cad, CancellationToken ct = default) =>
    RunEditAsync(cad, "solid.primitive",
        new PrimitiveParameters("box", new(0, 0, 0),
            Size: new Point3(80, 50, 20)), ct);
```

ICadJobApi.StartOperationAsync

計算ジョブを開始 — 切り離した形状を別プロセスで計算します。この時点では図面を変更しません。running／ready の保持は最大 4 件、同時実行は 1 件、時間制限は 1〜1800 秒です。

宣言

```
Task<JobInfo> StartOperationAsync(

    StartOperationRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	StartOperationRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。

引数	型	意味
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<JobInfo>。Task は await／Await で待機します。

プロパティ	型
Id	string
DocumentId	string
SourceRevision	long
Operation	string
State	string
Stage	string
ResultCount	int
ErrorCode	string
Message	string

要求型

```
public sealed record StartOperationRequest(OperationRequest Request, int TimeoutSeconds = 120);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadJobApi.GetJobAsync

計算の進捗を取得 — running、ready、completed、failed、cancelled、committed を区別します。ready はまだ図面へ反映されていません。

宣言

```
Task<JobInfo> GetJobAsync(

    string jobId,
```

```
    CancellationToken cancellationToken = default
);
```

引数	型	意味
jobId	string	StartOperationAsync で返った JobInfo.Id。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<JobInfo>。Task は await／Await で待機します。

プロパティ	型
Id	string
DocumentId	string
SourceRevision	long
Operation	string
State	string
Stage	string
ResultCount	int
ErrorCode	string
Message	string

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadJobApi.CancelJobAsync

計算や未確定結果を破棄 — running の計算を中止し、ready の未確定結果を破棄します。既に確定した図面の取消しには UndoAsync を使います。

宣言

```
Task<JobInfo> CancelJobAsync(
    string jobId,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
jobId	string	StartOperationAsync で返った JobInfo.Id。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<JobInfo>。Task は await／Await で待機します。

プロパティ	型
Id	string
DocumentId	string
SourceRevision	long
Operation	string
State	string
Stage	string
ResultCount	int
ErrorCode	string
Message	string

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

ICadJobApi.CommitJobAsync

計算結果を図面に反映 — ready の結果を 1 回の Undo で反映します。開始後に図面が変わっていれば拒否。開始とは別の UUID を使い、ライセンスも再判定します。

宣言

```
Task<EditResult> CommitJobAsync(
    CommitJobRequest request,
    CancellationToken cancellationToken = default
);
```

引数	型	意味
request	CommitJobRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<EditResult>。Task は await／Await で待機します。

プロパティ	型
Document	DocumentInfo
Entities	EntityInfo[]
Changed	bool

要求型

```
public sealed record CommitJobRequest(string DocumentId, long ExpectedRevision, string OperationId, string JobId);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

板金

SheetAsync は既定設定を QueryAsync で表示した後、例として板厚 1・内 R2・K 係数 0.4 の基本板を作ります。この例の数値を実部品の既定値として固定しないでください。後続のフランジなどは新しいエッジ参照を取り直して実行します。

コード例：sheet

```
public static async Task<EditResult> SheetAsync(
    ICadWorkspaceApi cad, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    QueryResult defaults = await cad.QueryAsync(new QueryRequest(
        doc.Id, doc.Revision, "sheetDefaults", Array.Empty<string>()), ct);
    Console.WriteLine(defaults.Data.GetRawText());
    EditResult profile = await RectangleAsync(cad, ct);
    // This example intentionally specifies t=1, R=2, K=0.4.
    // Use the drawing or the inspected project defaults for production parts.
    return await RunEditAsync(cad, "sheet.plate", new SheetPlateParameters(
        profile.Entities.Single().Id, new SheetRules(1, 2, .4)), ct);
}
```

計測と解析

計測・解析は worker-data です。completed 後に GetJobDataAsync で結果を取得します。AnalysisAsync は単一材料の学習例です。アセンブリは analysis.assembly.contacts／mesh／solve を使い、Body ごとに材料を指定します。節点・要素・境界を全ページ取得してから解析します。

コード例：data

```
public static async Task<JobDataPage> MeasureAsync(
    ICadWorkspaceApi cad, string[] entityIds, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    JobInfo job = await cad.StartOperationAsync(new StartOperationRequest(
        doc.Operation("measure.geometry", new MeasureParameters(entityIds))), ct);
    while (job.State == "running")
    {
        await Task.Delay(200, ct);
        job = await cad.GetJobAsync(job.Id, ct);
    }
    if (job.State != "completed")
        throw new InvalidOperationException($"{job.State}: {job.Message}");
    return await cad.GetJobDataAsync(new JobDataRequest(job.Id, "summary"), ct);
}
```

コード例：analysis

```
public static async Task<JobInfo> RunDataAsync<T>(
    ICadWorkspaceApi cad, string operation, T parameters,
    CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    JobInfo job = await cad.StartOperationAsync(new StartOperationRequest(
        doc.Operation(operation, parameters), TimeoutSeconds: 120), ct);
    while (job.State == "running")
    {
        await Task.Delay(200, ct);
        job = await cad.GetJobAsync(job.Id, ct);
    }
    if (job.State != "completed")
        throw new InvalidOperationException($"{job.State}: {job.Message}");
    return job;
}
public static async Task<T[]> ReadArrayAsync<T>(
    ICadWorkspaceApi cad, string jobId, string section,
    CancellationToken ct = default)
{
    var items = new List<T>();
    do
    {
        JobDataPage page = await cad.GetJobDataAsync(
            new JobDataRequest(jobId, section, items.Count, 128), ct);
        T[] batch = page.Data.Deserialize<T[]>(LocalProtocol.Json)
            ?? throw new InvalidOperationException("Missing result data.");
        items.AddRange(batch);
        if (items.Count >= page.Total || batch.Length == 0) break;
    } while (true);
    return items.ToArray();
}
public static async Task<JobDataPage> AnalysisAsync(
    ICadWorkspaceApi cad, CancellationToken ct = default)
{
    EditResult box = await BoxAsync(cad, ct);
    JobInfo job = await RunDataAsync(cad, "analysis.mesh",
        new AnalysisMeshParameters(new[] { box.Entities.Single().Id }, 3, 10), ct);
    double[][] points = await ReadArrayAsync<double[]>(cad, job.Id, "points", ct);
    int[][] elements = await ReadArrayAsync<int[]>(cad, job.Id, "elements", ct);
```

```
    AnalysisBoundaryData[] boundaries = await ReadArrayAsync<AnalysisBoundaryData>(
        cad, job.Id, "boundaries", ct);
    int[] GroupsAt(double x) => boundaries.Where(b =>
        b.Nodes.All(n => Math.Abs(points[n][0] - x) < 1e-6))
        .Select(b => b.Group).Distinct().ToArray();
    int[] fixedGroups = GroupsAt(0), loadedGroups = GroupsAt(80);
    if (fixedGroups.Length == 0 || loadedGroups.Length == 0)
        throw new InvalidOperationException("Expected end faces were not found.");
    var mesh = new AnalysisMeshData(3, points, elements, boundaries);
    var conditions = new[] {
        new AnalysisConditionData("fixed", Groups: fixedGroups),
        new AnalysisConditionData("totalforce", Groups: loadedGroups, Z: -100)
    };
    JobInfo result = await RunDataAsync(cad, "analysis.solve",
        new AnalysisSolveParameters(mesh, new AnalysisMaterialData(), conditions), ct);
    return await cad.GetJobDataAsync(new JobDataRequest(result.Id, "summary"), ct);
}
```

ICadWorkspaceApi.GetJobDataAsync

計測や解析の結果を取得 — completed のデータジョブから summary や結果配列をページ取得します。1〜256 件。データジョブには CommitJobAsync を呼びません。

宣言

```
Task<JobDataPage> GetJobDataAsync(

    JobDataRequest request,

    CancellationToken cancellationToken = default

);
```

引数	型	意味
request	JobDataRequest	下記の要求型を作成します。編集時は現在の図面 ID・revision と新しい UUID が必要です。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<JobDataPage>。Task は await／Await で待機します。

プロパティ	型
Job	JobInfo
Section	string
Data	Text.Json.JsonElement
Total	int
Offset	int

要求型

```
public sealed record JobDataRequest(string JobId,string Section="summary",int Offset=0,int Limit=128);
```

エラーと利用条件

図面取得には no_document、入力には invalid_argument、ID には document_changed／revision_conflict／entity_not_found を確認します。編集は cad_busy／layer_locked／undo_disabled を返す場合があります。権限が必要な操作は license_required です。スキーマ取得だけでは実行許可になりません。ジョブには job_not_found／job_not_ready、計算には geometry_failed／invalid_result 等があります。詳細は共通エラー表を参照してください。

レイアウト・注記・画面確認

LayoutAsync は A4 横に独立した平面ビューを配置します。紙面座標は紙の mm、ModelOrigin はモデルの WCS mm です。VisibleLayers は対象図形の実際のレイヤ名に合わせます。同名レイアウトがある場合は別名を使います。layout.annotations で文字付き引き出し線を追加し、view.capture で描画を確認できます。VIEWBASE／VIEWPROJ の対話的な投影関係とは別です。

コード例：layout

```
public static async Task<EditResult> LayoutAsync(
    ICadAutomationApi cad, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    var view = new LayoutViewParameters(new(0, 0, -1), new(0, 1, 0),
        new(40, 25, 0), new(100, 110, 0), 1,
        new LayoutRectangle(20, 180, 50, 180),
        new[] { "Standard", "0", "Example" });
    return await cad.ExecuteOperationAsync(doc.Operation("layout.create",
        new LayoutCreateParameters("Example-A4", 297, 210, new[] { view },
            Array.Empty<GeometryDefinition>(), Array.Empty<LayoutDimensionParameters>())), ct);
}
```

読み込み・保存・書き出し

絶対パスと形式を指定し、completed になるまで確認します。戻り値 Bytes・Sha256 でファイルを検証できます。既存ファイルの上書きは既定で拒否します。Save は図面名を変更し、Export はコピーを書き出します。Warnings も確認してください。document.open は未保存変更がない状態で使用します。保存・出力は形状の Undo では取り消せません。

コード例：files

```
public static async Task<FileOperationInfo> SaveAsync(
    ICadFileApi cad, string absolutePath, CancellationToken ct = default)
{
    FileFormatInfo[] formats = await cad.GetFileFormatsAsync(ct);
    if (!formats.Any(f => f.Format == "jdc" && f.CanSave && f.Available))
        throw new InvalidOperationException("JDC saving is unavailable.");
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    FileOperationInfo operation = await cad.SaveDocumentAsync(new SaveDocumentRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"), absolutePath, "jdc"), ct);
    return await WaitFileAsync(cad, operation, ct);
}
public static async Task<FileOperationInfo> ExportStepAsync(
    ICadFileApi cad, string absolutePath, CancellationToken ct = default)
{
    DocumentInfo doc = await cad.GetDocumentAsync(ct);
    FileOperationInfo operation = await cad.ExportDocumentAsync(new ExportDocumentRequest(
        doc.Id, doc.Revision, Guid.NewGuid().ToString("D"), absolutePath, "step"), ct);
    return await WaitFileAsync(cad, operation, ct);
}
public static async Task<FileOperationInfo> WaitFileAsync(
    ICadFileApi cad, FileOperationInfo operation, CancellationToken ct = default)
{
    try
    {
        while (operation.State == "running")
        {
            await Task.Delay(200, ct);
            operation = await cad.GetFileOperationAsync(operation.OperationId, ct);
        }
    }
    catch (OperationCanceledException)
    {
        await cad.CancelFileOperationAsync(operation.OperationId, CancellationToken.None);
        throw;
    }
    if (operation.State != "completed")
        throw new InvalidOperationException($"{operation.State}: {operation.Error?.Message}");
    foreach (string warning in operation.Warnings ?? Array.Empty<string>())
        Console.WriteLine(warning);
    Console.WriteLine($"{operation.FilePath}: {operation.Bytes} bytes; {operation.Sha256}");
    return operation;
}
```

ICadFileApi.GetFileFormatsAsync

```
Task<FileFormatInfo[]> GetFileFormatsAsync(CancellationToken cancellationToken = default);
```

完全な要求作成・状態確認・戻り値の使用例はこの分類の files コード例にあります。

ICadFileApi.SaveDocumentAsync

```
Task<FileOperationInfo> SaveDocumentAsync(SaveDocumentRequest request, CancellationToken cancellationToken = default);
```

完全な要求作成・状態確認・戻り値の使用例はこの分類の files コード例にあります。

ICadFileApi.ExportDocumentAsync

```
Task<FileOperationInfo> ExportDocumentAsync(ExportDocumentRequest request, CancellationToken cancellationToken =
default);
```

完全な要求作成・状態確認・戻り値の使用例はこの分類の files コード例にあります。

ICadFileApi.GetFileOperationAsync

```
Task<FileOperationInfo> GetFileOperationAsync(string operationId, CancellationToken cancellationToken = default);
```

完全な要求作成・状態確認・戻り値の使用例はこの分類の files コード例にあります。

ICadFileApi.CancelFileOperationAsync

```
Task<FileOperationInfo> CancelFileOperationAsync(string operationId, CancellationToken cancellationToken = default);
```

完全な要求作成・状態確認・戻り値の使用例はこの分類の files コード例にあります。

コマンド登録とアドイン

ExampleAddin と Samples を同じプロジェクトでビルドします。OnStartup でリボンを登録し、ボタンの ExecuteAsync から作図します。OnShutdown では自分で登録したイベントやタイマーを解除してください。この例は登録していないので処理を空にしています。独自アドインは対応ライセンスが必要です。

コード例：addin

```
public sealed class ExampleAddin : ICadAddin
{
 public void OnStartup(IAddinContext context)
 {
     context.RegisterTab(new RibbonTab("example", "Example", new[] {
         new RibbonPanel("draw", "Draw", new[] {
             new RibbonButton("line-circle", "Line and circle",
                 "Create example geometry", new DrawCommand())
         })
     }));
 }
 public void OnShutdown() { }
 private sealed class DrawCommand : IAddinCommand
 {
     public async Task<CommandResult> ExecuteAsync(ICadApi api, CancellationToken ct)
     {
         EditResult result = await Samples.DrawingAsync(api, ct);
         return new CommandResult($"Created {result.Entities.Length} objects.");
     }
 }
}
```

IAddinCommand.ExecuteAsync

追加ボタンの処理 — ボタンから呼ばれます。api で図面にアクセスし、CancellationToken を API へ渡します。返した Message が空でなければ本体が表示します。

宣言

```
Task<CommandResult> ExecuteAsync(

    ICadApi api,

    CancellationToken cancellationToken

);
```

引数	型	意味
api	ICadApi	本体の共通 API。必要な追加インターフェースへ is／TryCast で変換。
cancellationToken	CancellationToken	呼出しの中止要求。開始前の待機を中止できます。短い確定処理は開始後に完了して返ります。

戻り値

Task<CommandResult>。Task は await／Await で待機します。

プロパティ	型
Message	string

OnStartup での登録が成功してからタブが表示されます。Free 版は本体同梱サンプルだけです。自作・変更した DLL は STD／Pro で読み込んでください。

IAddinContext.RegisterTab

タブを登録 — OnStartup 内だけで呼べます。ID は重複不可。1 拡張あたり最大 10 タブ、1 タブ 1〜20 グループ、1 グループ 1〜50 ボタンです。

宣言

```
void RegisterTab(
    RibbonTab tab
);
```

引数	型	意味
tab	RibbonTab	タブ・グループ・ボタンを構成した RibbonTab。

戻り値

void

OnStartup での登録が成功してからタブが表示されます。Free 版は本体同梱サンプルだけです。自作・変更した DLL は STD／Pro で読み込んでください。

ICadAddin.OnStartup

起動時にタブを登録 — DLL 読込み後に本体から呼ばれます。context.RegisterTab でタブ・グループ・ボタンを登録します。重い計算や作図はボタン実行へ分けます。

宣言

```
void OnStartup(
    IAddinContext context
);
```

引数	型	意味
context	IAddinContext	本体が渡す登録コンテキスト。Api と AddinDirectory を参照可能。

戻り値

void

OnStartup での登録が成功してからタブが表示されます。Free 版は本体同梱サンプルだけです。自作・変更した DLL は STD／Pro で読み込んでください。

ICadAddin.OnShutdown

終了時に後処理 — 本体の終了時などに呼ばれます。自分で登録したイベントやリソースを解除します。図面を保存する操作はここへ隠さないでください。

宣言

```
void OnShutdown();
```

引数なし。

戻り値

void

OnStartup での登録が成功してからタブが表示されます。Free 版は本体同梱サンプルだけです。自作・変更した DLL は STD／Pro で読み込んでください。

メソッド名索引

CancelFileOperationAsync　550

CancelJobAsync　541

CommitJobAsync　543

CreateCurvesAsync　519

CreateGeometryAsync　518

DeleteEntitiesAsync　535

ExecuteAsync　551

ExecuteOperationAsync　516

ExportDocumentAsync　550

GetAttributesAsync　525

GetCapabilitiesAsync　513

GetDocumentAsync　512

GetEntitiesAsync　523

GetEntityDetailsAsync　524

GetFileFormatsAsync　549

GetFileOperationAsync　550

GetInteractiveCommandsAsync　531

GetJobAsync　540

GetJobDataAsync　546

GetLayersAsync　515

GetOperationSchemaAsync　514

GetSelectionAsync　522

GetTopologyAsync　526

LaunchInteractiveCommandAsync　532

OnShutdown　553

OnStartup　552

QueryAsync　527

RedoAsync　537

RegisterTab　552

SaveDocumentAsync　549

SetSelectionAsync　524

StartOperationAsync　539

TranslateEntitiesAsync　534

UndoAsync　536

93 操作のパラメーターリファレンス

共通の OperationRequest に操作名と Parameters を入れます。immediate は ExecuteOperationAsync、worker は StartOperationAsync→CommitJobAsync、worker-data は StartOperationAsync→GetJobDataAsync です。以下の必須欄は JSON 入力について示します。.NET コンストラクターの省略値・入れ子の型は続く公開型定義で確認できます。

設定・属性　557

作画・注記　560

編集・複写・ブロック　562

3D・面編集　568

板金　579

計測・解析　587

レイアウト・表示　593

ファイル　600

設定・属性 — 操作とパラメーター

styles.lineWidth　線幅定義を登録・更新

入力型：LineWidthParameters / immediate

同名の定義を更新し、使用中の図形・レイヤの参照を維持。Scale は layout（紙上 mm）/world（モデル mm）/device（画素）。

JSON 引数	必須	型
name	はい	string
width	はい	number
makeCurrent	省略可	boolean
scale	省略可	string

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("styles.lineWidth",
    new LineWidthParameters("0.5mm", 0.5, MakeCurrent: true), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

styles.linePattern　線種定義を登録・更新

入力型：LinePatternParameters / immediate

Pattern は線・隙間の正の長さを交互に指定。偶数個。空配列は実線。Scale は layout/world/device。

JSON 引数	必須	型
name	はい	string
pattern	はい	array of number
makeCurrent	省略可	boolean
scale	省略可	string

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("styles.linePattern",
    new LinePatternParameters("破線 4-2", [4, 2], MakeCurrent: true), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

attributes.set　図形のレイヤー・色・線種・線幅を変更

入力型：SetAttributesParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityIds	はい	array of string
layer	省略可	string / null
color	省略可	object / null
linePattern	省略可	string / null
lineWidth	省略可	string / null

```
public sealed record SetAttributesParameters(string[] EntityIds, string? Layer = null, CadColor? Color = null,
    string? LinePattern = null, string? LineWidth = null);
```

layers.delete　未使用レイヤーを削除

入力型：DeleteLayerParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string

```
public sealed record DeleteLayerParameters(string Name);
```

layers.set　レイヤー作成・属性変更・カレント設定

入力型：SetLayerParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string
color	省略可	object / null
locked	省略可	boolean / null
transparency	省略可	integer / null
makeCurrent	省略可	boolean
linePattern	省略可	string / null
lineWidth	省略可	string / null

```
public sealed record SetLayerParameters(string Name, CadColor? Color = null, bool? Locked = null,
    int? Transparency = null, bool MakeCurrent = false, string? LinePattern = null, string? LineWidth = null);
```

作画・注記 — 操作とパラメーター

annotation.dimension　点列・角度・半径・直径・座標・引出し寸法を作成

入力型：DimensionParameters / immediate / Free / STD / Pro

Kind: linear/coordinate/angle/radius/diameter/leader/baseline。点列による寸法。図形との新しい関連付けは未公開。

JSON 引数	必須	型
kind	はい	string
points	はい	array of object
textPosition	はい	object
normal	はい	object
direction	はい	object
radius	省略可	number
style	省略可	string / null

```
public sealed record DimensionParameters(string Kind,Point3[] Points,Point3 TextPosition,Point3 Normal,Point3 Direction,double Radius=0,string? Style=null);
```

annotation.hatch　閉輪郭から現在／指定スタイルのハッチを作成

入力型：HatchParameters / worker / Free / STD / Pro

単一の閉じた平面輪郭。Style 未指定は現在のハッチスタイル。複数境界・島の直接指定は未公開。

JSON 引数	必須	型
profileId	はい	string
style	省略可	string / null

```
public sealed record HatchParameters(string ProfileId,string? Style=null);
```

fill.create　閉輪郭の塗りつぶし・2 色グラデーションを一括作成

入力型：CreateFillsParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
fills	はい	array of object

```
public sealed record CreateFillsParameters(FillDefinition[] Fills);
```

geometry.create　点・線・円・円弧・ポリライン・楕円・スプライン・文字の作成

入力型：CreateObjectsParameters / immediate / Free / STD / Pro

GeometryDefinition は 1 種類だけ指定。色・レイヤーは現在値。

JSON 引数	必須	型
objects	はい	array of object

```
public sealed record CreateObjectsParameters(GeometryDefinition[] Objects);
```

image.import　Embed a raster image on a WCS plane

入力型：ImportImageParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
filePath	はい	string
location	はい	object
widthAxis	はい	object
heightAxis	はい	object

```
public sealed record ImportImageParameters(string FilePath, Point3 Location, Point3 WidthAxis, Point3 HeightAxis);
```

編集・複写・ブロック ― 操作とパラメーター

block.attributes　挿入ブロックの属性値を変更

入力型：BlockAttributesParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
values	はい	array of object

```
public sealed record BlockAttributesParameters(string EntityId,BlockValue[] Values);
```

block.create　選択図形から名前付きブロックを作成

入力型：BlockCreateParameters / immediate / Free / STD / Pro

定義名は一意。元の選択図形をブロック参照へ置換。

JSON 引数	必須	型
entityIds	はい	array of string
name	はい	string
basePoint	はい	object

```
public sealed record BlockCreateParameters(string[] EntityIds,string Name,Point3 BasePoint);
```

block.dynamic　既存ダイナミックブロックのパラメータを変更

入力型：BlockDynamicParameters / worker / Free / STD / Pro

既存の定義にある Key だけ変更。新しいパラメータ・アクション定義の作成は対話エディター。

JSON 引数	必須	型
entityId	はい	string
key	はい	string
value	はい	string

```
public sealed record BlockDynamicParameters(string EntityId,string Key,string Value);
```

block.insert　名前付きブロックを挿入

入力型：BlockInsertParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string
position	はい	object
scale	省略可	object / null
rotationDegrees	省略可	number
normal	省略可	object / null

```
public sealed record BlockInsertParameters(string Name,Point3 Position,Point3? Scale=null,double
RotationDegrees=0,Point3? Normal=null);
```

curve.join　接続した曲線列を 1 本のパスへ結合

入力型：CurveJoinParameters / worker / Free / STD / Pro

全入力が 1 本の接続パスとなる必要があります。

JSON 引数	必須	型
entityIds	はい	array of string
tolerance	省略可	number

```
public sealed record CurveJoinParameters(string[] EntityIds,double Tolerance=0.000001);
```

curve.offset　平面曲線の等距離線

入力型：CurveOffsetParameters / worker / Free / STD / Pro

指定 Normal を持つ平面上の曲線のみ。元の曲線は残ります。

JSON 引数	必須	型
entityId	はい	string
planeNormal	はい	object
distance	はい	number
tolerance	省略可	number

```
public sealed record CurveOffsetParameters(string EntityId,Point3 PlaneNormal,double Distance,double Tolerance=0.001);
```

curve.split　正規化パラメータで曲線を分割

入力型：CurveSplitParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
parameter	はい	number

```
public sealed record CurveSplitParameters(string EntityId,double Parameter);
```

curve.trim　曲線の始点・終点パラメータでトリム

入力型：TrimCurveParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
startParameter	はい	number
endParameter	はい	number

```
public sealed record TrimCurveParameters(string EntityId, double StartParameter, double EndParameter);
```

entities.array.circular　軸まわりの等角度複写

入力型：CircularArrayParameters / immediate / Free / STD / Pro

Count は作る複写の数。1×StepDegrees から配置。

JSON 引数	必須	型
entityIds	はい	array of string
center	はい	object
axis	はい	object
stepDegrees	はい	number
count	はい	integer

```
public sealed record CircularArrayParameters(string[] EntityIds, Point3 Center, Point3 Axis, double StepDegrees, int Count);
```

entities.array.linear　等間隔の複写

入力型：LinearArrayParameters / immediate / Free / STD / Pro

Count は作る複写の数。通常は 1×Step から。IncludeOriginalCopy=true では 0×Step から。

JSON 引数	必須	型
entityIds	はい	array of string
step	はい	object
count	はい	integer
includeOriginalCopy	省略可	boolean

```
public sealed record LinearArrayParameters(string[] EntityIds, Point3 Step, int Count, bool IncludeOriginalCopy = false);
```

entities.explode　パス・ブロック・シェル・ソリッドを 1 階層分解

入力型：ExplodeParameters / worker / Free / STD / Pro

1 階層の分解。入力は結果へ置換。ブロック属性や板金情報を保持する編集の代わりには使えません。

JSON 引数	必須	型
entityIds	はい	array of string

```
public sealed record ExplodeParameters(string[] EntityIds);
```

entities.mirror　平面を指定した鏡像

入力型：MirrorParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityIds	はい	array of string
planeOrigin	はい	object
planeNormal	はい	object
copy	省略可	boolean

```
public sealed record MirrorParameters(string[] EntityIds, Point3 PlaneOrigin, Point3 PlaneNormal, bool Copy = false);
```

entities.rotate　任意軸で回転・回転複写

入力型：RotateParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityIds	はい	array of string
center	はい	object
axis	はい	object
angleDegrees	はい	number
copy	省略可	boolean

```
public sealed record RotateParameters(string[] EntityIds, Point3 Center, Point3 Axis, double AngleDegrees, bool Copy = false);
```

entities.scale　基点を指定した等倍率の拡縮

入力型：ScaleParameters / immediate / Free / STD / Pro

正の等倍率のみ。認識済み板金は対象外。板金の寸法 API を利用。

JSON 引数	必須	型
entityIds	はい	array of string
center	はい	object
factor	はい	number
copy	省略可	boolean

```
public sealed record ScaleParameters(string[] EntityIds, Point3 Center, double Factor, bool Copy = false);
```

fill.set　塗りつぶしの色・グラデーション・角度・透明度を変更

入力型：SetFillParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
entityIds	はい	array of string
appearance	はい	object

```
public sealed record SetFillParameters(string[] EntityIds, FillAppearance Appearance);
```

geometry.update　図形の定義を変更（ID と属性を維持）

入力型：UpdateGeometryParameters / immediate / Free / STD / Pro

同じネイティブ型の定義へ更新し、図形 ID と属性を維持。任意の内部プロパティを書き換える機能ではありません。

JSON 引数	必須	型
updates	はい	array of object

```
public sealed record UpdateGeometryParameters(GeometryUpdate[] Updates);
```

text.update　文字内容だけを変更（書式・位置・ID を維持）

入力型：UpdateTextParameters / immediate / Free / STD / Pro

通常のトップレベル Text を 1〜100 個、文字内容だけ更新。1〜32768 文字。書式・配置・位置・ID を保持。ブロック属性定義は対象外。全件を検証して 1 回の Undo で反映。

JSON 引数	必須	型
updates	はい	array of object

```
public sealed record UpdateTextParameters(TextUpdate[] Updates);
```

3D・面編集 — 操作とパラメーター

face.label　面に文字・画像ラベルを貼付け／置換／削除。平面・曲面対応、穴と境界でクリップ。JDC へ埋め込み。形状・体積不変。ラベル一覧は cad_query kind=surfaceLabels。

入力型：SurfaceLabelParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
face	はい	object
center	はい	object
width	省略可	number
height	省略可	number
rotationDegrees	省略可	number
text	省略可	string / null
imagePath	省略可	string / null
fontName	省略可	string
color	省略可	string
mode	省略可	string
labelId	省略可	string / null

```
public sealed record SurfaceLabelParameters(SubobjectRef Face, Point3 Center, double Width=30, double Height=10,
    double RotationDegrees=0, string? Text=null, string? ImagePath=null, string FontName="Yu Gothic",
    string Color="000000", string Mode="add", string? LabelId=null);
```

face.move　指定面を移動

入力型：FaceMoveParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object
displacement	はい	object
moveConnected	省略可	boolean

```
public sealed record FaceMoveParameters(string EntityId,SubobjectRef[] Faces,Point3 Displacement,bool
MoveConnected=false);
```

face.offset　指定面のオフセット（通常コマンドと同じ経路）

入力型：FaceOffsetParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object
distance	はい	number

```
public sealed record FaceOffsetParameters(string EntityId, SubobjectRef[] Faces, double Distance);
```

face.radius　円柱等の半径を変更

入力型：FaceRadiusParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object
radius	はい	number

```
public sealed record FaceRadiusParameters(string EntityId,SubobjectRef[] Faces,double Radius);
```

face.remove　面削除・周辺面による修復

入力型：RemoveFacesParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object

```
public sealed record RemoveFacesParameters(string EntityId, SubobjectRef[] Faces);
```

face.rotate　指定面を回転

入力型：FaceRotateParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object
origin	はい	object
axis	はい	object
angleDegrees	はい	number
moveConnected	省略可	boolean

```
public sealed record FaceRotateParameters(string EntityId,SubobjectRef[] Faces,Point3 Origin,Point3 Axis,double AngleDegrees,bool MoveConnected=false);
```

geometry.extrude　閉輪郭／面の押出し、曲線の押出し面

入力型：ExtrudeParameters / worker / Free / STD / Pro

Solid=true は閉じた平面輪郭／面。false は曲線から側面を作成。DeleteSources=false が既定。

JSON 引数	必須	型
entityIds	はい	array of string
direction	はい	object
solid	省略可	boolean
deleteSources	省略可	boolean

```
public sealed record ExtrudeParameters(string[] EntityIds, Point3 Direction, bool Solid = true, bool DeleteSources = false);
```

geometry.clothoid　ネイティブのクロソイドを作成・更新

入力型：ClothoidParameters / immediate

ネイティブのクロソイド。A・弧長は mm。EntityId 指定の編集は図形 ID・属性を保持。geometry クエリで定義を取得可能。

JSON 引数	必須	型
first	はい	object
direction	はい	object
normal	はい	object

JSON 引数	必須	型
parameter	はい	number
startLength	はい	number
endLength	はい	number
clockwise	省略可	boolean
chordDirection	省略可	boolean
entityId	省略可	string / null

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("geometry.clothoid",
    new ClothoidParameters(new(0, 0, 0), new(1, 0, 0), new(0, 0, 1),
        Parameter: 100, StartLength: 0, EndLength: 100), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
// 既存クロソイドの編集は EntityId を指定。同じ図形 ID・属性を維持します。
```

geometry.clothoid.basic　対称クロソイド・円弧・クロソイドの基本型を作成

入力型：ClothoidBasicParameters / immediate

入口・交点 IP・出口の 2 接線方向と A・円弧 R から基本型を生成。成立しない A/R/交角は失敗。

JSON 引数	必須	型
entry	はい	object
intersection	はい	object
exit	はい	object
normal	はい	object
parameter	はい	number
radius	はい	number

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("geometry.clothoid.basic",
    new ClothoidBasicParameters(new(-100, 0, 0), new(0, 0, 0), new(0, 100, 0),
        new(0, 0, 1), Parameter: 20, Radius: 30), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

geometry.clothoid.stations　クロソイドの横断線／接線を作成

入力型：ClothoidStationParameters / immediate

Interval 指定は等間隔の横断線（最大 2000）。省略は 1 本。Tangent=true は 1 本の接線。

JSON 引数	必須	型
entityId	はい	string
start	はい	number
left	はい	number
right	はい	number
normal	はい	object
interval	省略可	number / null
tangent	省略可	boolean

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("geometry.clothoid.stations",
    new ClothoidStationParameters(ids[0], Start: 0, Left: 3, Right: 3,
        Normal: new(0, 0, 1), Interval: 10), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

geometry.helix　軸・半径・高さ・巻数でらせんパスを作成

入力型：HelixParameters / worker / Free / STD / Pro

らせんパス。axis は向きのみ。詳細は helix-interference.ja.html。

JSON 引数	必須	型
center	はい	object
axis	はい	object
baseRadius	はい	number
topRadius	はい	number
height	はい	number
turns	はい	number
segmentsPerTurn	省略可	integer

JSON 引数	必須	型
clockwise	省略可	boolean
startDirection	省略可	object / null

```
public sealed record HelixParameters(Point3 Center, Point3 Axis, double BaseRadius, double TopRadius,
    double Height, double Turns, int SegmentsPerTurn = 48, bool Clockwise = false, Point3? StartDirection = null);
```

geometry.loft　断面列と任意ガイドによるロフト

入力型：LoftParameters / worker / Free / STD / Pro

断面 2〜64。入力順がロフト順。GuideId は任意。

JSON 引数	必須	型
profileIds	はい	array of string
solid	省略可	boolean
guideId	省略可	string / null
deleteSources	省略可	boolean

```
public sealed record LoftParameters(string[] ProfileIds, bool Solid = true, string? GuideId = null, bool DeleteSources
= false);
```

geometry.revolve　断面を軸のまわりに回転

入力型：RevolveParameters / worker / Free / STD / Pro

角度は ±360 度、0 不可。軸は Origin と Axis。

JSON 引数	必須	型
entityId	はい	string
origin	はい	object
axis	はい	object
angleDegrees	省略可	number
deleteSource	省略可	boolean

```
public sealed record RevolveParameters(string EntityId, Point3 Origin, Point3 Axis, double AngleDegrees = 360, bool
DeleteSource = false);
```

geometry.screw　閉断面をピッチ・巻数指定でらせん回転しソリッドを作成

入力型：ScrewParameters / worker / Free / STD / Pro

JSON 引数	必須	型
profileId	はい	string
origin	はい	object
axis	はい	object
pitch	はい	number
turns	はい	number
rightHanded	省略可	boolean
deleteSource	省略可	boolean

```
public sealed record ScrewParameters(string ProfileId, Point3 Origin, Point3 Axis,
    double Pitch, double Turns, bool RightHanded = true, bool DeleteSource = false);
```

geometry.sweep　断面とパスによるスイープ

入力型：SweepParameters / worker / Free / STD / Pro

断面とパスの向き・形状は既存エンジンの条件に従います。

JSON 引数	必須	型
profileId	はい	string
pathId	はい	string
deleteSources	省略可	boolean

```
public sealed record SweepParameters(string ProfileId, string PathId, bool DeleteSources = false);
```

solid.boolean　ソリッドの和・差・積

入力型：BooleanParameters / worker / Free / STD / Pro

Kind は union/subtract/intersect。subtract の先頭が母材。入力ソリッドは結果と置換。空の結果は反映せず失敗。

JSON 引数	必須	型
entityIds	はい	array of string

JSON 引数	必須	型
kind	はい	string

```
public sealed record BooleanParameters(string[] EntityIds, string Kind);
```

solid.chamfer　基準面とエッジを指定した面取り

入力型：ChamferParameters / worker / Free / STD / Pro

ReferenceFace 側と反対側の距離。等辺面取りは Distance1=Distance2。

JSON 引数	必須	型
entityId	はい	string
referenceFace	はい	object
edges	はい	array of object
distance1	はい	number
distance2	はい	number

```
public sealed record ChamferParameters(string EntityId, SubobjectRef ReferenceFace, SubobjectRef[] Edges, double
Distance1, double Distance2);
```

solid.fillet.resize　指定フィレット面と連続する丸みの半径を変更

入力型：ResizeFilletParameters / worker

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object
radius	はい	number

solid.fillet　指定エッジのフィレット

入力型：FilletParameters / worker / Free / STD / Pro

単一シェルのソリッド。辺参照を指定。半径と形状により失敗します。

JSON 引数	必須	型
entityId	はい	string

JSON 引数	必須	型
edges	はい	array of object
radius	はい	number

```
public sealed record FilletParameters(string EntityId, SubobjectRef[] Edges, double Radius);
```

solid.fillet.remove　指定フィレット面を除去して支持面を復元

入力型：RemoveFacesParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object

```
public sealed record RemoveFacesParameters(string EntityId, SubobjectRef[] Faces);
```

solid.primitive　箱・円柱・円すい・球・トーラス

入力型：PrimitiveParameters / worker / Free / STD / Pro

JSON 引数	必須	型
kind	はい	string
origin	はい	object
size	省略可	object / null
axis	省略可	object / null
radius	省略可	number
secondRadius	省略可	number

```
public sealed record PrimitiveParameters(string Kind, Point3 Origin, Point3? Size = null, Point3? Axis = null,
    double Radius = 10, double SecondRadius = 5);
```

solid.shell　開口面と厚さを指定してシェル化

入力型：ShellParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
openFaces	はい	array of object
thickness	はい	number

```
public sealed record ShellParameters(string EntityId,SubobjectRef[] OpenFaces,double Thickness);
```

surface.create　閉じた平面輪郭から面を作成

入力型：CreateFaceParameters / worker / Free / STD / Pro

JSON 引数	必須	型
entityId	はい	string
deleteSource	省略可	boolean

```
public sealed record CreateFaceParameters(string EntityId, bool DeleteSource = false);
```

surface.sew　面とシェルを縫合

入力型：SewParameters / worker / Free / STD / Pro

入力の面／シェルを縫合結果へ置換。

JSON 引数	必須	型
entityIds	はい	array of string

```
public sealed record SewParameters(string[] EntityIds);
```

surface.trim　単独サーフェスを別面で分割

入力型：SurfaceTrimParameters / worker / Free / STD / Pro

対象は単独の Face。両側の分割結果を返します。ソリッド内の面の部分トリムはこの API の対象外。

JSON 引数	必須	型
target	はい	object
cutter	はい	object

```
public sealed record SurfaceTrimParameters(SubobjectRef Target, SubobjectRef Cutter);
```

板金 — 操作とパラメーター

sheet.bend　線を指定して曲げる

入力型：SheetBendParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
lineId	はい	string
fixedFace	はい	object
rules	はい	object
angleDegrees	省略可	number
positionReference	省略可	integer

```
public sealed record SheetBendParameters(string EntityId, string LineId, SubobjectRef FixedFace, SheetRules Rules,
    double AngleDegrees = 90, int PositionReference = 0);
```

sheet.contour　連続フランジを作成

入力型：SheetContourParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
edge	はい	object
rules	はい	object
steps	はい	array of object

```
public sealed record SheetContourParameters(string EntityId, SubobjectRef Edge, SheetRules Rules, SheetStep[] Steps);
```

sheet.corner　コーナーを処理

入力型：SheetCornerParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
faces	はい	array of object

JSON 引数	必須	型
gap	はい	number
kind	省略可	integer

```
public sealed record SheetCornerParameters(string EntityId, SubobjectRef[] Faces, double Gap, int Kind = 0);
```

sheet.editBend　曲げ角・半径・長さを編集

入力型：SheetEditBendParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
bendFace	はい	object
angleDegrees	はい	number
radius	はい	number
length	はい	number
lengthReference	省略可	integer

```
public sealed record SheetEditBendParameters(string EntityId, SubobjectRef BendFace, double AngleDegrees,
    double Radius, double Length, int LengthReference = 0);
```

sheet.editForm　成形の編集・削除

入力型：SheetEditFormParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
formId	はい	string
dimensions	はい	object
delete	省略可	boolean

```
public sealed record SheetEditFormParameters(string EntityId, string FormId, SheetFormDimensions Dimensions, bool
Delete = false);
```

sheet.flange　エッジフランジを作成

入力型：SheetFlangeParameters / worker / STD / Pro

StartDistance／EndDistance でフランジの開始・終了範囲を指定できます。端部ごとのテーパーは共通 API 未公開です。寸法基準と既定値は SheetFlangeParameters を参照してください。

JSON 引数	必須	型
entityId	はい	string
edges	はい	array of object
rules	はい	object
length	はい	number
angleDegrees	省略可	number
gap	省略可	number
autoCorner	省略可	boolean
positionReference	省略可	integer
lengthReference	省略可	integer
startDistance	省略可	number / null
endDistance	省略可	number / null

```
public sealed record SheetFlangeParameters(string EntityId, SubobjectRef[] Edges, SheetRules Rules,
    double Length, double AngleDegrees = 90, double Gap = .1, bool AutoCorner = true,
    int PositionReference = 0, int LengthReference = 0, double? StartDistance = null, double? EndDistance = null);
```

sheet.flatPattern　輪郭・曲げ線・注記から展開図を作成

入力型：SheetFlatPatternParameters / worker / STD / Pro

展開図の図形をモデルへ追加。ファイルへの DXF 等の書出しではありません。

JSON 引数	必須	型
entityId	はい	string
placement	はい	object
bendLines	省略可	boolean
annotations	省略可	boolean
bendTable	省略可	boolean

```
public sealed record SheetFlatPatternParameters(string EntityId, Point3 Placement, bool BendLines = true, bool
Annotations = true, bool BendTable = false);
```

sheet.form　バーリング・ダボ・エンボス・ビード・ルーバー・絞り・ブリッジを作成

入力型：SheetFormParameters / worker / STD / Pro

成形の寸法は SheetFormDimensions。ルーバー／ブリッジのゼロ幅切込みは展開図の SM_CUT へ。製造時の伸び代は計算しません。

JSON 引数	必須	型
entityId	はい	string
face	はい	object
center	はい	object
dimensions	はい	object
rotationDegrees	省略可	number
reverse	省略可	boolean

```
public sealed record SheetFormParameters(string EntityId, SubobjectRef Face, Point3 Center, SheetFormDimensions
Dimensions,
    double RotationDegrees = 0, bool Reverse = false);
```

sheet.hem　開きヘムを作成

入力型：SheetHemParameters / worker / STD / Pro

開きヘム。閉じた潰しヘムではありません。

JSON 引数	必須	型
entityId	はい	string
edges	はい	array of object
rules	はい	object
length	はい	number
gap	はい	number

```
public sealed record SheetHemParameters(string EntityId, SubobjectRef[] Edges, SheetRules Rules, double Length, double
Gap);
```

sheet.jog　段曲げを作成

入力型：SheetJogParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
edge	はい	object
rules	はい	object
height	はい	number
length	はい	number
angleDegrees	省略可	number

```
public sealed record SheetJogParameters(string EntityId, SubobjectRef Edge, SheetRules Rules, double Height, double Length, double AngleDegrees = 90);
```

sheet.loft　ロール・ロフト板金の作成／寸法・角 R 編集

入力型：SheetLoftParameters / worker / STD / Pro

EntityId 未指定で新規、指定すると既存ロフト寸法を編集。加工を保持できる条件は本体と同じ。

JSON 引数	必須	型
rules	はい	object
dimensions	はい	object
entityId	省略可	string / null

```
public sealed record SheetLoftParameters(SheetRules Rules, SheetLoftDimensions Dimensions, string? EntityId = null);
```

sheet.manageBend　曲げの抑制・復元・削除・親変更

入力型：SheetManageBendParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
bendIndex	はい	integer
operation	はい	integer
parentIndex	省略可	integer

```
public sealed record SheetManageBendParameters(string EntityId, int BendIndex, int Operation, int ParentIndex = -1);
```

sheet.plate　板金基本板を作成

入力型：SheetPlateParameters / worker / STD / Pro

JSON 引数	必須	型
profileId	はい	string
rules	はい	object
thicknessSide	省略可	integer

```
public sealed record SheetPlateParameters(string ProfileId, SheetRules Rules, int ThicknessSide = 0);
```

sheet.recognize　既存ソリッドを板金認識

入力型：SheetRecognizeParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
baseFace	はい	object
rules	はい	object
detectThickness	省略可	boolean

```
public sealed record SheetRecognizeParameters(string EntityId, SubobjectRef BaseFace, SheetRules Rules, bool
DetectThickness = true);
```

sheet.refold　加工を保持して再曲げ

入力型：SheetBodyParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string

```
public sealed record SheetBodyParameters(string EntityId);
```

sheet.relief　曲げ逃げを作成／除去

入力型：SheetReliefParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
edges	はい	array of object
width	はい	number
depth	はい	number
round	省略可	boolean
remove	省略可	boolean

```
public sealed record SheetReliefParameters(string EntityId, SubobjectRef[] Edges, double Width, double Depth, bool Round = false, bool Remove = false);
```

sheet.rip　指定エッジに切れ目を作成

入力型：SheetRipParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string
edges	はい	array of object
gap	はい	number
side	省略可	integer

```
public sealed record SheetRipParameters(string EntityId, SubobjectRef[] Edges, double Gap, int Side = 2);
```

sheet.rules　既存板金の板厚・曲げ設定を変更

入力型：SheetRulesParameters / worker / STD / Pro

厚さ・半径・K 係数・逃げ寸法。展開補正表の全定義は画面版を利用。

JSON 引数	必須	型
entityId	はい	string
rules	はい	object

```
public sealed record SheetRulesParameters(string EntityId, SheetRules Rules);
```

sheet.unfold　板金を展開

入力型：SheetBodyParameters / worker / STD / Pro

JSON 引数	必須	型
entityId	はい	string

```
public sealed record SheetBodyParameters(string EntityId);
```

計測・解析 — 操作とパラメーター

analysis.mesh　単一材料の 2D／3D メッシュを生成

入力型：AnalysisMeshParameters / worker-data / STD / Pro

2D の閉輪郭または 3D の 1 ソリッド。解析用ネイティブランタイムと JDesignKernelWorker も必要。

JSON 引数	必須	型
entityIds	はい	array of string
dimension	はい	integer
meshSize	省略可	number
tolerance	省略可	number

```
public sealed record AnalysisMeshParameters(string[] EntityIds,int Dimension,double MeshSize=5,double Tolerance=.01);
```

analysis.assembly.contacts　複数ソリッド間の接合面を検出

入力型：AssemblyContactsParameters / worker-data

閉じた単一シェルのソリッド 2〜100 個。体積の重なりは不可。接合面は完全接合または非接合で、摩擦・接触離脱を伴う非線形接触ではありません。

JSON 引数	必須	型
entityIds	はい	array of string
tolerance	省略可	number

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("analysis.assembly.contacts",
    new AssemblyContactsParameters(ids), selectResult: false);
var job = await cad.StartOperationAsync(new(request, 180), ct);
while (job.State == "running")
{
    await Task.Delay(100, ct);
    job = await cad.GetJobAsync(job.Id, ct);
}
if (job.State != "completed") throw new InvalidOperationException(job.Message);
var result = await cad.GetJobDataAsync(new(job.Id, "summary"), ct);
// 配列はセクション名と Offset/Limit を指定して全ページを取得します。
// contacts セクションに接合面の Group、Body1/Body2、面積、輪郭を返します。
```

analysis.assembly.mesh　部品別領域・接合設定を保持した 3D メッシュを生成

入力型：AssemblyMeshParameters / worker-data

節点・要素のほか elementBodies/bodyNames/contacts/bodySurfaces を全ページ取得。部品番号は入力 ID の順で 1 始まり。接合設定を変えたらメッシュを作り直します。

JSON 引数	必須	型
entityIds	はい	array of string
meshSize	省略可	number
tolerance	省略可	number
excludedContactGroups	省略可	array / null

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("analysis.assembly.mesh",
    new AssemblyMeshParameters(ids, MeshSize: 5, ExcludedContactGroups: []), selectResult: false);
var job = await cad.StartOperationAsync(new(request, 180), ct);
while (job.State == "running")
{
    await Task.Delay(100, ct);
    job = await cad.GetJobAsync(job.Id, ct);
}
if (job.State != "completed") throw new InvalidOperationException(job.Message);
var result = await cad.GetJobDataAsync(new(job.Id, "summary"), ct);
// 配列はセクション名と Offset/Limit を指定して全ページを取得します。
// Body 番号は ids の順に 1 始まり。検出時と同じ順序で渡します。
// 接合を解除する面の Group を ExcludedContactGroups に指定します。
```

analysis.assembly.solve　部品ごとの材料でアセンブリの応力／熱解析

入力型：AssemblySolveParameters / worker-data

BodyMaterials は BodyNames と同じ順。弾性・定常／非定常熱解析。メッシュ接続情報を維持して入力。結果はデータ取得、解析画面のスタディ登録は未公開。

JSON 引数	必須	型
mesh	はい	object
bodyMaterials	はい	array of object
conditions	はい	array of object
kind	省略可	string

JSON 引数	必須	型
transientThermal	省略可	boolean
initialTemperature	省略可	number
endTime	省略可	number
timeStep	省略可	number
outputCount	省略可	integer

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
// assemblyMesh は assembly.mesh の全ページから復元した AssemblyMeshData。
// leftNodes/rightNodes はそのメッシュの境界節点番号(int[])、0 始まりです。
// BodyMaterials は BodyNames と同じ順序・件数です。
var request = doc.Operation("analysis.assembly.solve",
    new AssemblySolveParameters(assemblyMesh,
        [new AnalysisMaterialData("低熱伝導材", Conductivity: 10),
         new AnalysisMaterialData("高熱伝導材", Conductivity: 30)],
        [new AnalysisConditionData("temperature", Target: "node", NodeIds: leftNodes, X: 0),
         new AnalysisConditionData("temperature", Target: "node", NodeIds: rightNodes, X: 100)],
        Kind: "heat"), selectResult: false);
var job = await cad.StartOperationAsync(new(request, 180), ct);
while (job.State == "running")
{
    await Task.Delay(100, ct);
    job = await cad.GetJobAsync(job.Id, ct);
}
if (job.State != "completed") throw new InvalidOperationException(job.Message);
var result = await cad.GetJobDataAsync(new(job.Id, "summary"), ct);
// 配列はセクション名と Offset/Limit を指定して全ページを取得します。
```

analysis.solve　メッシュ・材料・境界条件を指定し応力／定常・非定常熱解析

入力型：AnalysisSolveParameters / worker-data / STD / Pro

単一材料。メッシュ・条件・材料を明示。結果を解析画面のスタディへ登録する機能は未公開。

JSON 引数	必須	型
mesh	はい	object
material	はい	object
conditions	はい	array of object
kind	省略可	string
thickness	省略可	number
planeStrain	省略可	boolean

JSON 引数	必須	型
transientThermal	省略可	boolean
initialTemperature	省略可	number
endTime	省略可	number
timeStep	省略可	number
outputCount	省略可	integer

```
public sealed record AnalysisSolveParameters(AnalysisMeshData Mesh,AnalysisMaterialData
Material,AnalysisConditionData[] Conditions,string Kind="elastic",double Thickness=1,bool PlaneStrain=false,
    bool TransientThermal=false,double InitialTemperature=20,double EndTime=60,double TimeStep=1,int OutputCount=61);
```

measure.contour　複数曲線から閉領域を認識して周長・面積・図心・断面二次モーメントを取得

入力型：ContourMeasureParameters / worker-data

明示した候補曲線から種となる 1 辺につながる閉輪郭を認識。分岐や隙間は失敗として返します。元図形を変更せず、summary に周長・面積・図心・図心軸の断面二次モーメントと使用した ID を返します。

JSON 引数	必須	型
seedId	はい	string
candidateIds	はい	array of string
origin	はい	object
normal	はい	object
xDirection	はい	object

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("measure.contour",
    new ContourMeasureParameters(ids[0], ids,
        new(0, 0, 0), new(0, 0, 1), new(1, 0, 0)), selectResult: false);
var job = await cad.StartOperationAsync(new(request, 180), ct);
while (job.State == "running")
{
    await Task.Delay(100, ct);
    job = await cad.GetJobAsync(job.Id, ct);
}
if (job.State != "completed") throw new InvalidOperationException(job.Message);
```

```
var result = await cad.GetJobDataAsync(new(job.Id, "summary"), ct);
// 配列はセクション名と Offset/Limit を指定して全ページを取得します。
```

measure.highlight　認識した閉輪郭を一時ハイライト／解除

入力型：ContourHighlightParameters / immediate

モデル空間で認識した輪郭線を一時表示。Clear=true で解除。Undo・保存対象を増やしません。

JSON 引数	必須	型
contour	省略可	object / null
clear	省略可	boolean

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("measure.highlight",
    new ContourHighlightParameters(new ContourMeasureParameters(ids[0], ids,
        new(0, 0, 0), new(0, 0, 1), new(1, 0, 0))), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
// 解除: 同じ操作名に new ContourHighlightParameters(Clear: true) を渡します。
```

measure.geometry　曲線長・閉鎖・平面性・断面積・体積・重心を計算

入力型：MeasureParameters / worker-data / Free / STD / Pro

summary/items で長さ・閉鎖・平面性・断面積・体積・重心。対象外型は status=unsupported。慣性は断面のローカル平面軸に対する値。

JSON 引数	必須	型
entityIds	はい	array of string
tolerance	省略可	number

```
public sealed record MeasureParameters(string[] EntityIds,double Tolerance=0.001);
```

solid.interference　ソリッド間の干渉・接触を形状を変更せずに検査

入力型：InterferenceParameters / worker-data / Free / STD / Pro

静止したソリッドの干渉・面接触。summary/pairs を取得。failed を干渉なしと解釈しない。

JSON 引数	必須	型
entityIds	はい	array of string

JSON 引数	必須	型
tolerance	省略可	number
pairs	省略可	array / null

```
public sealed record InterferenceParameters(string[] EntityIds, double Tolerance = 0.000001, InterferencePair[]? Pairs
= null);
```

レイアウト・表示 — 操作とパラメーター

layout.activate　作成済みレイアウトを表示（以後のモデル操作にはモデル表示へ戻す）

入力型：LayoutActivateParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string

```
public sealed record LayoutActivateParameters(string Name);
```

layout.annotations　既存紙面の注釈を文字付き引き出し線へ置換／追加。矢印先端と文字の棚位置を指定。製図ルール: R・穴数・長穴・面取り等の注記は、独立した文字と線ではなく、原則として文字付き引き出し線を使用します。円・円弧の位置やピッチを示す寸法は、意図に応じて正確な中心を基準とし、同じ数値になる接点で代用しません。必要に応じ中心マーク・中心線を付けます。全長・接線・端面基準の寸法は図面の指定に従い、ユーザーが現状維持を指定した寸法は変更しません。

入力型：LayoutAnnotationsParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string
removeIndices	はい	array of integer
leaders	はい	array of object

```
public sealed record LayoutAnnotationsParameters(string Name,int[] RemoveIndices,LayoutLeaderParameters[] Leaders);
```

layout.view　既存レイアウトのビューを追加・更新・削除

入力型：LayoutViewEditParameters / immediate

Name は既存レイアウト名。add は Index=-1、update/remove は layoutViews で取得した Index。更新はビューの識別情報と用紙注記を保持します。

JSON 引数	必須	型
name	はい	string
mode	はい	string
index	省略可	integer

JSON 引数	必須	型
view	省略可	object / null

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var layouts = await cad.QueryAsync(new(doc.Id, doc.Revision, "layouts", []), ct);
string layoutName = layouts.Data[0].GetProperty("name").GetString()!;
// クリップ範囲は用紙内に、VisibleLayers は実在するレイヤ名に合わせます。
var request = doc.Operation("layout.view",
    new LayoutViewEditParameters(layoutName, "add", View:
        new LayoutViewParameters(new(0, 0, -1), new(0, 1, 0),
            new(0, 0, 0), new(60, 60, 0), 1,
            new(10, 110, 10, 110), ["Standard"], "hiddenLines")), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
// 更新は Mode="update", Index を指定。削除は Mode="remove", View=null。
// Index と現状は QueryAsync(kind:"layoutViews") で取得してください。
```

section.plane　断面平面の作成・位置／方向／反転・ライブ断面設定

入力型：SectionPlaneParameters / immediate

EntityId 未指定で作成、指定で同じ断面平面を更新。Live・Flipped・Active は別々に設定。法線と XDirection は直交。

JSON 引数	必須	型
origin	はい	object
normal	はい	object
xDirection	はい	object
width	省略可	number
height	省略可	number
live	省略可	boolean
flipped	省略可	boolean
active	省略可	boolean
entityId	省略可	string / null

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("section.plane",
    new SectionPlaneParameters(new(0, 0, 5), new(0, 0, 1), new(1, 0, 0),
        Width: 100, Height: 75, Live: true, Flipped: false), selectResult: false);
```

```
var result = await cad.ExecuteOperationAsync(request, ct);
// result.Entities[0].Id が平面の ID。編集時は EntityId に渡します。
// 反転は Flipped=true、ライブ表示の解除は Live=false。
```

section.generate　指定図形と断面平面から XY 断面図を生成

入力型：SectionGenerateParameters / worker

PlaneId と対象 ID を明示。XY 上の InsertionPoint へ 2D 断面を追加。元の 3D 図形と断面平面は保持。

JSON 引数	必須	型
planeId	はい	string
entityIds	はい	array of string
insertionPoint	はい	object

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var planes = await cad.QueryAsync(new(doc.Id, doc.Revision, "sectionPlanes", []), ct);
string planeId = planes.Data[0].GetProperty("entity").GetProperty("id").GetString()!;
var request = doc.Operation("section.generate",
    new SectionGenerateParameters(planeId,
        ids.Where(id => id != planeId).ToArray(), new(150, 0, 0)), selectResult: false);
var job = await cad.StartOperationAsync(new(request, 180), ct);
while (job.State == "running")
{
    await Task.Delay(100, ct);
    job = await cad.GetJobAsync(job.Id, ct);
}
if (job.State != "ready") throw new InvalidOperationException(job.Message);
doc = await cad.GetDocumentAsync(ct);
var result = await cad.CommitJobAsync(new(doc.Id, doc.Revision,
    Guid.NewGuid().ToString(), job.Id), ct);
```

animation.view　アニメーションビューの作成・設定・削除

入力型：AnimationViewParameters / immediate

標準のアニメーションビュー。繰返し・速度・方向・表示レイヤを設定し JDC 保存可能。Mode=remove で削除。

JSON 引数	必須	型
name	はい	string
direction	はい	object
up	はい	object
repeatMode	省略可	string

JSON 引数	必須	型
repeatCount	省略可	integer
speed	省略可	number
visibleLayers	省略可	array / null
mode	省略可	string

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("animation.view",
    new AnimationViewParameters("動作ビュー", new(1, 1, -1), new(0, 0, 1),
        RepeatMode: "count", RepeatCount: 3, Speed: 1), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

animation.drive　回転・曲線・2 曲線ドライブの作成・更新・削除

入力型：AnimationDriveParameters / immediate

回転は度、曲線は曲線始点からの mm。dualCurve は 2 本の曲線ドライブに従います。循環依存は不可。削除前に参照するドライブ／スケジュールを解除。

JSON 引数	必須	型
name	はい	string
kind	はい	string
entityIds	はい	array of string
axisOrigin	省略可	object / null
axisDirection	省略可	object / null
curveId	省略可	string / null
firstDrive	省略可	string / null
secondDrive	省略可	string / null
dependsOn	省略可	string / null
position	省略可	number
nullPosition	省略可	number
tangential	省略可	boolean
stretch	省略可	boolean

JSON 引数	必須	型
mode	省略可	string

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var selection = await cad.GetSelectionAsync(cancellationToken: ct);
string[] ids = selection.Entities.Select(e => e.Id).ToArray();
// ID は今開いている図面から取得します。保存・再起動後は取り直してください。
var request = doc.Operation("animation.drive",
    new AnimationDriveParameters("回転", "axis", ids,
        AxisOrigin: new(0, 0, 0), AxisDirection: new(0, 0, 1)), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
// 曲線ドライブは Kind="curve", CurveId=曲線の ID。
// 2 曲線は Kind="dualCurve", FirstDrive/SecondDrive=曲線ドライブ名。
```

animation.schedule　アニメーションの動作時間・位置を設定／削除

入力型：AnimationScheduleParameters / immediate

秒と位置のキーをドライブごとに指定。各トラック 2 個以上の異なる時刻。曲線端連動は ToCurveEnd=true。設定変更前に停止。

JSON 引数	必須	型
name	はい	string
tracks	はい	array of object
mode	省略可	string

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("animation.schedule",
    new AnimationScheduleParameters("動作",
        [new AnimationTrack("回転", [new(0, 0), new(5, 90), new(10, 0)])]), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
```

animation.control　アニメーションの再生・一時停止・再開・停止

入力型：AnimationControlParameters / immediate

play/pause/resume/stop。再生姿勢は表示用で元図形を変形しません。ビューの設定を変更する場合は先に stop。

JSON 引数	必須	型
view	はい	string

JSON 引数	必須	型
action	はい	string
schedule	省略可	string / null
start	省略可	number
end	省略可	number / null
speed	省略可	number / null

C#例。api は IAddinContext.Api、ct は CancellationToken。図面の選択・名称・寸法を指定して実行します。

```
// using JDesignCAD.Api; using System.Linq;
var cad = (ICadWorkspaceApi)api;
var doc = await cad.GetDocumentAsync(ct);
var request = doc.Operation("animation.control",
    new AnimationControlParameters("動作ビュー", "play", Schedule: "動作"), selectResult: false);
var result = await cad.ExecuteOperationAsync(request, ct);
// Action="pause" / "resume" / "stop"。設定変更前に停止してください。
```

layout.create　投影図・紙面図形・実寸表示の寸法を持つレイアウトを作成

入力型：LayoutCreateParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
name	はい	string
width	はい	number
height	はい	number
views	はい	array of object
objects	はい	array of object
dimensions	はい	array of object

```
public sealed record LayoutCreateParameters(string Name,double Width,double Height,LayoutViewParameters[] Views,
    GeometryDefinition[] Objects,LayoutDimensionParameters[] Dimensions);
```

view.activate　モデル／レイアウトの表示を切り替え。倍率を維持（zoomToFit 指定時だけフィット）。ビュー名は cad_query kind=views で取得。図形・Undo は変更しません。

入力型：ActivateViewParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
kind	はい	string

JSON 引数	必須	型
name	省略可	string / null
zoomToFit	省略可	boolean

```
public sealed record ActivateViewParameters(string Kind, string? Name = null, bool ZoomToFit = false);
```

view.capture　モデル／指定レイアウトを PNG へ描画して確認（図形変更なし）

入力型：DrawingCaptureParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
filePath	はい	string
layoutName	省略可	string / null
width	省略可	integer
height	省略可	integer
overwrite	省略可	boolean

```
public sealed record DrawingCaptureParameters(string FilePath,string? LayoutName=null,int Width=1600,int
Height=1100,bool Overwrite=false);
```

ファイル — 操作とパラメーター

document.open　既存 JDC をこのウィンドウで開く。未保存の変更があれば拒否。Undo 対象外。図面未表示時は documentId 空文字・expectedRevision=0。タイムアウト後は同じ operationId で再照会。

入力型：OpenDocumentParameters / immediate / Free / STD / Pro

JSON 引数	必須	型
filePath	はい	string

```
public sealed record OpenDocumentParameters(string FilePath);
```

公開データ型の正式定義

C#の宣言を公開ソースから抽出しています。=以降は省略時の値、?は null を許容する型、[]は配列です。JSON Schema にある必須指定も併せて確認してください。レコードの名称はメソッドの要求型・戻り値と対応します。

ActivateViewParameters

```
public sealed record ActivateViewParameters(string Kind, string? Name = null, bool ZoomToFit = false);
```

AnalysisBoundaryData

```
public sealed record AnalysisBoundaryData(int Group,int[] Nodes,int Body=0);
```

AnalysisConditionData

```
public sealed record AnalysisConditionData(string Kind,int Group=0,int Node=-1,string Target="boundary",int[]?
Groups=null,int[]? NodeIds=null,int[]? BoundaryIndices=null,double X=0,double Y=0,double Z=0);
```

AnalysisMaterialData

```
public sealed record AnalysisMaterialData(string Name="Steel",double YoungModulus=200000,double PoissonRatio=.3,double
Density=7850,double Conductivity=50,double SpecificHeat=460,double ExpansionMicro=12);
```

AnalysisMeshData

```
public sealed record AnalysisMeshData(int Dimension,double[][] Points,int[][] Elements,AnalysisBoundaryData[]
Boundaries,AnalysisBoundaryData[]? EdgeBoundaries=null,int[]? VertexNodes=null);
```

AnalysisMeshParameters

```
public sealed record AnalysisMeshParameters(string[] EntityIds,int Dimension,double MeshSize=5,double Tolerance=.01);
```

AnalysisSolveParameters

```
public sealed record AnalysisSolveParameters(AnalysisMeshData Mesh,AnalysisMaterialData
Material,AnalysisConditionData[] Conditions,string Kind="elastic",double Thickness=1,bool PlaneStrain=false,
    bool TransientThermal=false,double InitialTemperature=20,double EndTime=60,double TimeStep=1,int OutputCount=61);
```

ArcDefinition

```
public sealed record ArcDefinition(Point3 Center, Point3 Start, Point3 Normal, double SweepDegrees);
```

AttributeInfo

```
public sealed record AttributeInfo(EntityInfo Entity, CadColor? Color, string? LinePattern, string? LineWidth,
    bool LayerLocked);
```

AttributePage

```
public sealed record AttributePage(DocumentInfo Document, AttributeInfo[] Items);
```

BlockAttributesParameters

```
public sealed record BlockAttributesParameters(string EntityId,BlockValue[] Values);
```

BlockCreateParameters

```
public sealed record BlockCreateParameters(string[] EntityIds,string Name,Point3 BasePoint);
```

BlockDynamicParameters

```
public sealed record BlockDynamicParameters(string EntityId,string Key,string Value);
```

BlockInsertParameters

```
public sealed record BlockInsertParameters(string Name,Point3 Position,Point3? Scale=null,double
RotationDegrees=0,Point3? Normal=null);
```

BlockValue

```
public sealed record BlockValue(string Name,string Value);
```

BooleanParameters

```
public sealed record BooleanParameters(string[] EntityIds, string Kind);
```

Bounds3

```
public sealed record Bounds3(Point3 Minimum, Point3 Maximum);
```

CadColor

```
public sealed record CadColor(string Mode, string? Rgb = null);
```

CapabilityInfo

```
public sealed record CapabilityInfo(int Major, int Minor, string CoordinateSystem, string LengthUnit,
    OperationInfo[] Operations, string[] Limitations)
{
    public LicenseCapabilities? License { get; init; }
    public FileFormatInfo[]? FileFormats { get; init; }
    /// <summary>Input methods are separate from geometry operations; points are supplied by the user in CAD.</summary>
    public string[]? InputMethods { get; init; }
}
```

ChamferParameters

```
public sealed record ChamferParameters(string EntityId, SubobjectRef ReferenceFace, SubobjectRef[] Edges, double
Distance1, double Distance2);
```

CircleDefinition

```
public sealed record CircleDefinition(Point3 Center, double Radius, Point3 Normal);
```

CircularArrayParameters

```
public sealed record CircularArrayParameters(string[] EntityIds, Point3 Center, Point3 Axis, double StepDegrees, int
Count);
```

CommandResult

```
public sealed record CommandResult(string Message);
```

CommitJobRequest

```
public sealed record CommitJobRequest(string DocumentId, long ExpectedRevision, string OperationId, string JobId);
```

CreateCurvesRequest

```
public sealed record CreateCurvesRequest(string DocumentId, long ExpectedRevision, string OperationId,
    PolylineDefinition[] Polylines, ArcDefinition[] Arcs, bool SelectCreated = true);
```

CreateFaceParameters

```
public sealed record CreateFaceParameters(string EntityId, bool DeleteSource = false);
```

CreateFillsParameters

```
public sealed record CreateFillsParameters(FillDefinition[] Fills);
```

CreateGeometryRequest

```
public sealed record CreateGeometryRequest(string DocumentId, long ExpectedRevision, string OperationId,
    LineDefinition[] Lines, CircleDefinition[] Circles, bool SelectCreated = true);
```

CreateObjectsParameters

```
public sealed record CreateObjectsParameters(GeometryDefinition[] Objects);
```

CurveJoinParameters

```
public sealed record CurveJoinParameters(string[] EntityIds,double Tolerance=0.000001);
```

CurveOffsetParameters

```
public sealed record CurveOffsetParameters(string EntityId,Point3 PlaneNormal,double Distance,double Tolerance=0.001);
```

CurveSplitParameters

```
public sealed record CurveSplitParameters(string EntityId,double Parameter);
```

DeleteLayerParameters

```
public sealed record DeleteLayerParameters(string Name);
```

DimensionParameters

```
public sealed record DimensionParameters(string Kind,Point3[] Points,Point3 TextPosition,Point3 Normal,Point3
Direction,double Radius=0,string? Style=null);
```

DocumentInfo

```
public sealed record DocumentInfo(string Id, string? FileName, long Revision, int ObjectCount,
    bool CanUndo, bool CanRedo, string LengthUnit = "mm", string CoordinateSystem = "WCS");
```

DocumentRequest

```
public sealed record DocumentRequest(string DocumentId, long ExpectedRevision);
```

DrawingCaptureParameters

```
public sealed record DrawingCaptureParameters(string FilePath,string? LayoutName=null,int Width=1600,int
Height=1100,bool Overwrite=false);
```

EditResult

```
public sealed record EditResult(DocumentInfo Document, EntityInfo[] Entities, bool Changed);
```

EllipseDefinition

```
public sealed record EllipseDefinition(Point3 Center, Point3 MajorAxis, Point3 MinorAxis,
    double StartDegrees = 0, double SweepDegrees = 360);
```

EntityDetail

```
public sealed record EntityDetail(EntityInfo Entity, bool LayerLocked, string GeometryStatus,
    LineDefinition? Line = null, CircleDefinition? Circle = null, ArcDefinition? Arc = null, PolylineDefinition?
Polyline = null);
```

EntityDetailsRequest

```
public sealed record EntityDetailsRequest(string DocumentId, long ExpectedRevision, string[] EntityIds);
```

EntityDetailsResult

```
public sealed record EntityDetailsResult(DocumentInfo Document, EntityDetail[] Entities);
```

EntityEditRequest

```
public sealed record EntityEditRequest(string DocumentId, long ExpectedRevision, string OperationId, string[]
EntityIds);
```

EntityInfo

```
public sealed record EntityInfo(string Id, string Kind, string? Layer, Bounds3? Bounds);
```

EntityPage

```
public sealed record EntityPage(DocumentInfo Document, EntityInfo[] Entities, int Total, int Offset);
```

EntityQuery

```
public sealed record EntityQuery(string DocumentId, long ExpectedRevision, int Offset = 0, int Limit = 256);
```

ExplodeParameters

```
public sealed record ExplodeParameters(string[] EntityIds);
```

ExportDocumentRequest

```
public sealed record ExportDocumentRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string FilePath, string Format, bool Overwrite = false, string[]? EntityIds = null,
    Point3? BasePoint = null, IfcFileOptions? Ifc = null, int TimeoutSeconds = 300);
```

ExtrudeParameters

```
public sealed record ExtrudeParameters(string[] EntityIds, Point3 Direction, bool Solid = true, bool DeleteSources =
false);
```

FaceMoveParameters

```
public sealed record FaceMoveParameters(string EntityId,SubobjectRef[] Faces,Point3 Displacement,bool
MoveConnected=false);
```

FaceOffsetParameters

```
public sealed record FaceOffsetParameters(string EntityId, SubobjectRef[] Faces, double Distance);
```

FaceRadiusParameters

```
public sealed record FaceRadiusParameters(string EntityId,SubobjectRef[] Faces,double Radius);
```

FaceRotateParameters

```
public sealed record FaceRotateParameters(string EntityId,SubobjectRef[] Faces,Point3 Origin,Point3 Axis,double
AngleDegrees,bool MoveConnected=false);
```

FileFormatInfo

```
public sealed record FileFormatInfo(string Format, string[] Extensions, bool CanSave, bool CanExport,
    bool Available, string Description, string? UnavailableCode = null, LicenseRestriction? LicenseRestriction = null);
```

FileOperationInfo

```
public sealed record FileOperationInfo(string OperationId, string DocumentId, long SourceRevision,
    string Kind, string FilePath, string Format, string State, bool DocumentSaved = false,
    long? Bytes = null, string? Sha256 = null, string[]? Warnings = null, FileOutputError? Error = null)
{
    /// <summary>OBJ's uniquely named MTL sidecar. Keep these files beside the main output when transferring
it.</summary>
    public FileOutputArtifact[] AdditionalFiles { get; init; } = [];
}
```

FileOutputArtifact

```
public sealed record FileOutputArtifact(string FilePath, long Bytes, string Sha256);
```

FileOutputError

```
public sealed record FileOutputError(string Code, string Message, bool Retryable = false,
    LicenseRestriction? LicenseRestriction = null);
```

FillAppearance

```
public sealed record FillAppearance(CadColor Color, string Pattern = "none",
    CadColor? Color2 = null, double AngleDegrees = 0, int Transparency = 0);
```

FillDefinition

```
public sealed record FillDefinition(GeometryDefinition Boundary, FillAppearance Appearance, string? Layer = null);
```

FilletParameters

```
public sealed record FilletParameters(string EntityId, SubobjectRef[] Edges, double Radius);
```

GeometryDefinition

```
public sealed record GeometryDefinition(LineDefinition? Line = null, CircleDefinition? Circle = null,
    ArcDefinition? Arc = null, PolylineDefinition? Polyline = null, EllipseDefinition? Ellipse = null,
    SplineDefinition? Spline = null, TextDefinition? Text = null, Point3? Point = null);
```

GeometryUpdate

```
public sealed record GeometryUpdate(string EntityId, GeometryDefinition Geometry);
```

HatchParameters

```
public sealed record HatchParameters(string ProfileId,string? Style=null);
```

HelixParameters

```
public sealed record HelixParameters(Point3 Center, Point3 Axis, double BaseRadius, double TopRadius,
    double Height, double Turns, int SegmentsPerTurn = 48, bool Clockwise = false, Point3? StartDirection = null);
```

HistoryRequest

```
public sealed record HistoryRequest(string DocumentId, long ExpectedRevision, string OperationId);
```

IfcFileOptions

```
public sealed record IfcFileOptions(string Output = "ifc4Advanced", string? Name = null);
```

ImportImageParameters

```
public sealed record ImportImageParameters(string FilePath, Point3 Location, Point3 WidthAxis, Point3 HeightAxis);
```

InteractiveCommandInfo

```
public sealed record InteractiveCommandInfo(string Id,string Group,bool Available,string? UnavailableReason=null)
{
    public string? UnavailableCode { get; init; }
    public LicenseRestriction? LicenseRestriction { get; init; }
}
```

InteractiveCommandRequest

```
public sealed record InteractiveCommandRequest(string DocumentId,long ExpectedRevision,string OperationId,string
CommandId);
```

InteractiveCommandResult

```
public sealed record InteractiveCommandResult(string CommandId,string State,string Message);
```

InterferencePair

```
public sealed record InterferencePair(string FirstEntityId, string SecondEntityId);
```

InterferenceParameters

```
public sealed record InterferenceParameters(string[] EntityIds, double Tolerance = 0.000001, InterferencePair[]? Pairs
= null);
```

JobDataPage

```
public sealed record JobDataPage(JobInfo Job,string Section,JsonElement Data,int Total,int Offset);
```

JobDataRequest

```
public sealed record JobDataRequest(string JobId,string Section="summary",int Offset=0,int Limit=128);
```

JobInfo

```
public sealed record JobInfo(string Id, string DocumentId, long SourceRevision, string Operation, string State,
    string Stage, int ResultCount = 0, string? ErrorCode = null, string? Message = null);
```

LayerInfo

```
public sealed record LayerInfo(string Name, CadColor Color, string? LinePattern, string? LineWidth,
    bool Locked, int Transparency, bool Current);
```

LayerPage

```
public sealed record LayerPage(DocumentInfo Document, LayerInfo[] Layers, string[] LinePatterns, string[] LineWidths);
```

LayoutActivateParameters

```
public sealed record LayoutActivateParameters(string Name);
```

LayoutAnnotationsParameters

```
public sealed record LayoutAnnotationsParameters(string Name,int[] RemoveIndices,LayoutLeaderParameters[] Leaders);
```

LayoutCreateParameters

```
public sealed record LayoutCreateParameters(string Name,double Width,double Height,LayoutViewParameters[] Views,
    GeometryDefinition[] Objects,LayoutDimensionParameters[] Dimensions);
```

LayoutDimensionParameters

```
public sealed record LayoutDimensionParameters(DimensionParameters Dimension,double ValueScale=1,double
TextHeight=2.5);
```

LayoutLeaderParameters

```
public sealed record LayoutLeaderParameters(string Text,Point3 Anchor,Point3 Landing,double TextHeight=3,
    string Font="Yu Gothic",Point3[]? Elbows=null,CadColor? Color=null);
```

LayoutRectangle

```
public sealed record LayoutRectangle(double Left,double Right,double Bottom,double Top);
```

LayoutViewParameters

```
public sealed record LayoutViewParameters(Point3 Direction,Point3 Up,Point3 ModelOrigin,Point3 PaperOrigin,
    double Scale,LayoutRectangle Clip,string[] VisibleLayers,string Display="hiddenLines");
```

LicenseCapabilities

```
public sealed record LicenseCapabilities(string CurrentPlan, bool CustomAddinsAllowed, bool McpAvailable = true,
string FreeAddinPolicy = "bundled_sample_only");
```

LicenseRestriction

```
public sealed record LicenseRestriction(string Feature, string Operation, string CurrentPlan, string[] RequiredPlans,
bool Retryable = false);
```

LineDefinition

```
public sealed record LineDefinition(Point3 Start, Point3 End);
```

LinearArrayParameters

```
public sealed record LinearArrayParameters(string[] EntityIds, Point3 Step, int Count, bool IncludeOriginalCopy =
false);
```

LoftParameters

```
public sealed record LoftParameters(string[] ProfileIds, bool Solid = true, string? GuideId = null, bool DeleteSources
= false);
```

MeasureParameters

```
public sealed record MeasureParameters(string[] EntityIds,double Tolerance=0.001);
```

MirrorParameters

```
public sealed record MirrorParameters(string[] EntityIds, Point3 PlaneOrigin, Point3 PlaneNormal, bool Copy = false);
```

OpenDocumentParameters

```
public sealed record OpenDocumentParameters(string FilePath);
```

OperationInfo

```
public sealed record OperationInfo(string Name, string Description, string ParameterType, string Execution,
    bool Available, string? UnavailableReason = null)
{
    public string? UnavailableCode { get; init; }
    public LicenseRestriction? LicenseRestriction { get; init; }
}
```

OperationRequest

```
public sealed record OperationRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string Operation, JsonElement Parameters, bool SelectResult = true);
```

OperationSchema

```
public sealed record OperationSchema(OperationInfo Operation, JsonElement ParametersSchema);
```

Point3

```
public sealed record Point3(double X, double Y, double Z);
```

PolylineDefinition

```
public sealed record PolylineDefinition(Point3[] Vertices, bool Closed = false);
```

PrimitiveParameters

```
public sealed record PrimitiveParameters(string Kind, Point3 Origin, Point3? Size = null, Point3? Axis = null,
    double Radius = 10, double SecondRadius = 5);
```

QueryRequest

```
public sealed record QueryRequest(string DocumentId,long ExpectedRevision,string Kind,string[] EntityIds);
```

QueryResult

```
public sealed record QueryResult(DocumentInfo Document,JsonElement Data);
```

RemoveFacesParameters

```
public sealed record RemoveFacesParameters(string EntityId, SubobjectRef[] Faces);
```

RevolveParameters

```
public sealed record RevolveParameters(string EntityId, Point3 Origin, Point3 Axis, double AngleDegrees = 360, bool
DeleteSource = false);
```

RibbonButton

```
public sealed record RibbonButton(string Id, string Text, string Description, IAddinCommand Command,
    bool RequiresDocument = true, string? SmallIconPath = null, string? LargeIconPath = null);
```

RibbonPanel

```
public sealed record RibbonPanel(string Id, string Text, RibbonButton[] Buttons);
```

RibbonTab

```
public sealed record RibbonTab(string Id, string Text, RibbonPanel[] Panels);
```

RotateParameters

```
public sealed record RotateParameters(string[] EntityIds, Point3 Center, Point3 Axis, double AngleDegrees, bool Copy =
false);
```

SaveDocumentRequest

```
public sealed record SaveDocumentRequest(string DocumentId, long ExpectedRevision, string OperationId,
```

```
    string FilePath, string Format, bool Overwrite = false, int TimeoutSeconds = 300);
```

ScaleParameters

```
public sealed record ScaleParameters(string[] EntityIds, Point3 Center, double Factor, bool Copy = false);
```

ScrewParameters

```
public sealed record ScrewParameters(string ProfileId, Point3 Origin, Point3 Axis,
    double Pitch, double Turns, bool RightHanded = true, bool DeleteSource = false);
```

SelectionInfo

```
public sealed record SelectionInfo(DocumentInfo Document, EntityInfo[] Entities, int Total, int Offset);
```

SetAttributesParameters

```
public sealed record SetAttributesParameters(string[] EntityIds, string? Layer = null, CadColor? Color = null,
    string? LinePattern = null, string? LineWidth = null);
```

SetFillParameters

```
public sealed record SetFillParameters(string[] EntityIds, FillAppearance Appearance);
```

SetLayerParameters

```
public sealed record SetLayerParameters(string Name, CadColor? Color = null, bool? Locked = null,
    int? Transparency = null, bool MakeCurrent = false, string? LinePattern = null, string? LineWidth = null);
```

SewParameters

```
public sealed record SewParameters(string[] EntityIds);
```

SheetBendParameters

```
public sealed record SheetBendParameters(string EntityId, string LineId, SubobjectRef FixedFace, SheetRules Rules,
    double AngleDegrees = 90, int PositionReference = 0);
```

SheetBodyParameters

```
public sealed record SheetBodyParameters(string EntityId);
```

SheetContourParameters

```
public sealed record SheetContourParameters(string EntityId, SubobjectRef Edge, SheetRules Rules, SheetStep[] Steps);
```

SheetCornerParameters

```
public sealed record SheetCornerParameters(string EntityId, SubobjectRef[] Faces, double Gap, int Kind = 0);
```

SheetEditBendParameters

```
public sealed record SheetEditBendParameters(string EntityId, SubobjectRef BendFace, double AngleDegrees,
    double Radius, double Length, int LengthReference = 0);
```

SheetEditFormParameters

```
public sealed record SheetEditFormParameters(string EntityId, string FormId, SheetFormDimensions Dimensions, bool
Delete = false);
```

SheetFlangeParameters

```
public sealed record SheetFlangeParameters(string EntityId, SubobjectRef[] Edges, SheetRules Rules,
    double Length, double AngleDegrees = 90, double Gap = .1, bool AutoCorner = true,
    int PositionReference = 0, int LengthReference = 0, double? StartDistance = null, double? EndDistance = null);
```

SheetFlatPatternParameters

```
public sealed record SheetFlatPatternParameters(string EntityId, Point3 Placement, bool BendLines = true, bool
Annotations = true, bool BendTable = false);
```

SheetFormDimensions

```
public sealed record SheetFormDimensions(int Kind, double Diameter = 6, double Height = 4, double RootRadius = 1,
    double CrownRadius = 2, double Span = 10, double PilotDiameter = 4, bool Suppressed = false)
{
    // Additive property preserves the existing positional constructor and DLL ABI.
    public double CornerRadius { get; init; } = 1;
    public double BridgeHoleDiameter { get; init; }
    /// <summary>Rectangular draw flank angle to the original sheet plane, degrees. Default 90 preserves existing
forms.</summary>
    public double WallAngleDegrees { get; init; } = 90;
}
```

SheetFormParameters

```
public sealed record SheetFormParameters(string EntityId, SubobjectRef Face, Point3 Center, SheetFormDimensions
Dimensions,
    double RotationDegrees = 0, bool Reverse = false);
```

SheetHemParameters

```
public sealed record SheetHemParameters(string EntityId, SubobjectRef[] Edges, SheetRules Rules, double Length, double
Gap);
```

SheetJogParameters

```
public sealed record SheetJogParameters(string EntityId, SubobjectRef Edge, SheetRules Rules, double Height, double
Length, double AngleDegrees = 90);
```

SheetLoftDimensions

```
public sealed record SheetLoftDimensions(int Kind = 1, int FormingMode = 1, double BottomRadius = 30, double TopRadius
= 20,
    double Height = 50, double BottomWidth = 80, double BottomDepth = 60, double TopWidth = 50, double TopDepth = 40,
    double CornerRadius = 5, double TopCornerRadius = 5, double OffsetX = 0, double OffsetY = 0,
    double SeamGap = 1, int Segments = 32, int SeamMode = 0, double SeamAngle = 0, double Angle = 350,
    Point3? Origin = null);
```

SheetLoftParameters

```
public sealed record SheetLoftParameters(SheetRules Rules, SheetLoftDimensions Dimensions, string? EntityId = null);
```

SheetManageBendParameters

```
public sealed record SheetManageBendParameters(string EntityId, int BendIndex, int Operation, int ParentIndex = -1);
```

SheetPlateParameters

```
public sealed record SheetPlateParameters(string ProfileId, SheetRules Rules, int ThicknessSide = 0);
```

SheetRecognizeParameters

```
public sealed record SheetRecognizeParameters(string EntityId, SubobjectRef BaseFace, SheetRules Rules, bool
DetectThickness = true);
```

SheetReliefParameters

```
public sealed record SheetReliefParameters(string EntityId, SubobjectRef[] Edges, double Width, double Depth, bool
Round = false, bool Remove = false);
```

SheetRipParameters

```
public sealed record SheetRipParameters(string EntityId, SubobjectRef[] Edges, double Gap, int Side = 2);
```

SheetRules

```
public sealed record SheetRules(double Thickness = 1, double Radius = 1, double KFactor = .4,
    int ReliefType = 0, double ReliefWidth = 1, double ReliefDepth = 2);
```

SheetRulesParameters

```
public sealed record SheetRulesParameters(string EntityId, SheetRules Rules);
```

SheetStep

```
public sealed record SheetStep(double Length, double AngleDegrees, double Radius);
```

ShellParameters

```
public sealed record ShellParameters(string EntityId,SubobjectRef[] OpenFaces,double Thickness);
```

SplineDefinition

```
public sealed record SplineDefinition(Point3[] ThroughPoints, int Degree = 3, bool Closed = false);
```

StartOperationRequest

```
public sealed record StartOperationRequest(OperationRequest Request, int TimeoutSeconds = 120);
```

SubobjectRef

```
public sealed record SubobjectRef(string EntityId, long Revision, string Kind, int Index);
```

SurfaceLabelParameters

```
public sealed record SurfaceLabelParameters(SubobjectRef Face, Point3 Center, double Width=30, double Height=10,
    double RotationDegrees=0, string? Text=null, string? ImagePath=null, string FontName="Yu Gothic",
    string Color="000000", string Mode="add", string? LabelId=null);
```

SurfaceTrimParameters

```
public sealed record SurfaceTrimParameters(SubobjectRef Target, SubobjectRef Cutter);
```

SweepParameters

```
public sealed record SweepParameters(string ProfileId, string PathId, bool DeleteSources = false);
```

TextDefinition

```
public sealed record TextDefinition(string Content, Point3 Location, Point3 XDirection, Point3 YDirection,
    double Height, string Font = "Arial");
```

TextUpdate

```
public sealed record TextUpdate(string EntityId, string Content);
```

TopologyItem

```
public sealed record TopologyItem(SubobjectRef Reference, string GeometryType, Bounds3? Bounds,
    Point3? Position = null, Point3? Normal = null, double? Length = null);
```

TopologyPage

```
public sealed record TopologyPage(DocumentInfo Document, TopologyItem[] Items, int Total, int Offset);
```

TopologyRequest

```
public sealed record TopologyRequest(string DocumentId, long ExpectedRevision, string EntityId,
    string Kind = "face", int Offset = 0, int Limit = 128);
```

TranslateEntitiesRequest

```
public sealed record TranslateEntitiesRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string[] EntityIds, Point3 Displacement, bool Copy = false, bool SelectResult = true);
```

TrimCurveParameters

```
public sealed record TrimCurveParameters(string EntityId, double StartParameter, double EndParameter);
```

UpdateGeometryParameters

```
public sealed record UpdateGeometryParameters(GeometryUpdate[] Updates);
```

UpdateTextParameters

```
public sealed record UpdateTextParameters(TextUpdate[] Updates);
```

AnalysisContactData

```
public sealed record AnalysisContactData(int Group,int Body1,int Body2,double Area,bool Enabled=true,double[][]?
Outline=null);
```

AnimationControlParameters

```
public sealed record AnimationControlParameters(string View, string Action, string? Schedule=null,
    double Start=0, double? End=null, double? Speed=null);
```

AnimationDriveParameters

```
public sealed record AnimationDriveParameters(string Name, string Kind, string[] EntityIds,
    Point3? AxisOrigin=null, Point3? AxisDirection=null, string? CurveId=null,
    string? FirstDrive=null, string? SecondDrive=null, string? DependsOn=null,
    double Position=0, double NullPosition=0, bool Tangential=true, bool Stretch=false, string Mode="upsert");
```

AnimationKeyframe

```
public sealed record AnimationKeyframe(double Time, double Position, bool ToCurveEnd=false);
```

AnimationScheduleParameters

```
public sealed record AnimationScheduleParameters(string Name, AnimationTrack[] Tracks, string Mode="upsert");
```

AnimationTrack

```
public sealed record AnimationTrack(string Drive, AnimationKeyframe[] Keys);
```

AnimationViewParameters

```
public sealed record AnimationViewParameters(string Name, Point3 Direction, Point3 Up,
    string RepeatMode="once", int RepeatCount=1, double Speed=1, string[]? VisibleLayers=null, string Mode="upsert");
```

AssemblyContactsParameters

```
public sealed record AssemblyContactsParameters(string[] EntityIds, double Tolerance=.01);
```

AssemblyMeshData

```
public sealed record AssemblyMeshData(AnalysisMeshData Mesh,int[] ElementBodies,string[] BodyNames,
    AnalysisContactData[] Contacts,AnalysisBoundaryData[] BodySurfaces);
```

AssemblyMeshParameters

```
public sealed record AssemblyMeshParameters(string[] EntityIds, double MeshSize=5, double Tolerance=.01, int[]?
ExcludedContactGroups=null);
```

AssemblySolveParameters

```
public sealed record AssemblySolveParameters(AssemblyMeshData Mesh,AnalysisMaterialData[] BodyMaterials,
    AnalysisConditionData[] Conditions,string Kind="elastic",bool TransientThermal=false,
    double InitialTemperature=20,double EndTime=60,double TimeStep=1,int OutputCount=61);
```

CadInputRequest

```
public sealed record CadInputRequest(string DocumentId, long ExpectedRevision, string OperationId,
    string Kind, string Prompt, PointInputOptions? PointOptions = null,
    SelectionInputOptions? SelectionOptions = null, ValueInputOptions? ValueOptions = null, bool AllowNone = false);
```

CadInputResult

```
public sealed record CadInputResult(string InputId, string Kind, string State, DocumentInfo Document,
    Point3? Point = null, double? Number = null, string? Text = null, EntityInfo[]? Entities = null,
    Point3? PickPoint = null, Point3? CursorPoint = null, string? Reason = null)
{
    public bool Accepted => State == "accepted";
    /// <summary>Face/edge picks from a top-level body, with the same index/revision convention as GetTopologyAsync.
    /// Entities contains the owning bodies. Subobject selection uses clicks, not window/preselection.</summary>
    public SubobjectRef[]? Subobjects { get; init; }
}
```

ClothoidBasicParameters

```
public sealed record ClothoidBasicParameters(Point3 Entry, Point3 Intersection, Point3 Exit,
    Point3 Normal, double Parameter, double Radius);
```

ClothoidParameters

```
public sealed record ClothoidParameters(Point3 First, Point3 Direction, Point3 Normal,
    double Parameter, double StartLength, double EndLength, bool Clockwise = false, bool ChordDirection = false,
string? EntityId = null);
```

ClothoidStationParameters

```
public sealed record ClothoidStationParameters(string EntityId, double Start, double Left, double Right,
    Point3 Normal, double? Interval = null, bool Tangent = false);
```

ContourHighlightParameters

```
public sealed record ContourHighlightParameters(ContourMeasureParameters? Contour = null, bool Clear = false);
```

ContourMeasureParameters

```
public sealed record ContourMeasureParameters(string SeedId, string[] CandidateIds,
    Point3 Origin, Point3 Normal, Point3 XDirection);
```

InputPreviewRequest

```
public sealed record InputPreviewRequest(string InputId, LineDefinition[] Lines, CircleDefinition[] Circles);
```

LayoutViewEditParameters

```
public sealed record LayoutViewEditParameters(string Name, string Mode, int Index = -1, LayoutViewParameters? View =
null);
```

LinePatternParameters

```
public sealed record LinePatternParameters(string Name, double[] Pattern, bool MakeCurrent = false, string Scale =
"layout");
```

LineWidthParameters

```
public sealed record LineWidthParameters(string Name, double Width, bool MakeCurrent = false, string Scale =
"layout");
```

PointInputOptions

```
public sealed record PointInputOptions(Point3? BasePoint = null, bool RubberBand = true);
```

ResizeFilletParameters

```
public sealed record ResizeFilletParameters(string EntityId, SubobjectRef[] Faces, double Radius);
```

SectionGenerateParameters

```
public sealed record SectionGenerateParameters(string PlaneId, string[] EntityIds, Point3 InsertionPoint);
```

SectionPlaneParameters

```
public sealed record SectionPlaneParameters(Point3 Origin, Point3 Normal, Point3 XDirection,
    double Width = 100, double Height = 75, bool Live = false, bool Flipped = false, bool Active = true, string?
EntityId = null);
```

SelectionFilter

```
public sealed record SelectionFilter(string[]? Kinds = null, string[]? Layers = null, bool IncludeLocked = false);
```

SelectionInputOptions

```
public sealed record SelectionInputOptions(SelectionFilter? Filter = null, bool UsePreselection = false,
    int MinimumCount = 1, int MaximumCount = 1000, string Target = "entity");
```

SelectionRegionRequest

```
public sealed record SelectionRegionRequest(string DocumentId, long ExpectedRevision, string Mode,
    Point3? First = null, Point3? Second = null, SelectionFilter? Filter = null, bool SetSelection = true);
```

ValueInputOptions

```
public sealed record ValueInputOptions(string? DefaultText = null, double? DefaultNumber = null,
    double? Minimum = null, double? Maximum = null, string[]? Keywords = null, bool AllowSpaces = true);
```