Chrome DevTools クイックリファレンス

要素、コンソール、ネットワーク、パフォーマンス、デバッグ

要素

検査と編集

右クリック -> 検査	要素の Elements パネルを開く
タグ/属性をダブルクリック	HTML をインラインで編集
Delete キー	選択ノードを削除
Ctrl+Z	DOM の変更を元に戻す
H キー	選択要素の表示を切り替え
ノードをドラッグ	DOM ツリーで要素を移動

スタイルパネル

element.style {}	要素にインラインスタイルを追加
プロパティ値をクリック	CSS 値をライブで編集
ルール横のチェックボックス	CSS プロパティのオン/オフを切り替え
:hov ボタン	疑似状態を強制（:hover、:focus）
.cls ボタン	CSS クラスを追加/削除
カラースウォッチ	カラーピッカーを開く
Computed タブ	最終的な計算済み CSS 値を表示

コンソール

Console API

```
console.log("info");          console.warn("warning");
console.error("error");       console.table(arrayOrObj);
console.group("label");       console.groupEnd();
console.time("t"); /*...*/ console.timeEnd("t");
```

コンソールユーティリティ

$0	Elements で現在選択中の要素
$('sel') / $$('sel')	querySelector / querySelectorAll の短縮形
copy(obj)	オブジェクトを文字列としてクリップボードにコピー
clear()	コンソール出力をクリア
monitor(fn)	関数 fn の呼び出しをログ記録
monitorEvents(el, 'click')	要素のイベントをログ記録
keys(obj) / values(obj)	オブジェクトのキー / 値
$_	最後に評価した式の結果

フィルタリング

ログレベルのドロップダウン	verbose/info/warn/error でフィルタ
フィルターテキストボックス	コンソール出力を検索
-url:extension	ソース URL でメッセージを除外
context: ドロップダウン	iframe/worker コンテキストでフィルタ

ネットワーク

ネットワークパネルの機能

フィルターバー	種別でフィルタ: XHR、JS、CSS、Img、Doc、WS
検索（Ctrl+F）	すべてのリクエスト/レスポンス本文を横断検索
ログを保持	ページ遷移してもログを保持
キャッシュを無効化	DevTools を開いている間ブラウザキャッシュをバイパス
スロットリングのドロップダウン	低速 3G、高速 3G、オフラインをシミュレート
リクエスト URL をブロック	右クリックでリクエスト→URL をブロック

リクエスト詳細タブ

Headers	リクエスト/レスポンスヘッダー、ステータスコード
Payload	POST 本文、クエリパラメーター
Preview	フォーマット済みレスポンス（JSON、HTML、画像）
Response	生のレスポンス本文
Timing	DNS、接続、TLS、TTFB、ダウンロード
Initiator	リクエストを発生させたスタックトレース
Cookies	送受信された Cookie

コピーとエクスポート

右クリック -> cURL としてコピー	リクエストを cURL コマンドとしてコピー
fetch としてコピー	fetch() JavaScript としてコピー
HAR をエクスポート	すべてのリクエストを HAR ファイルとしてエクスポート
レスポンスをコピー	レスポンス本文をクリップボードにコピー

ソース

ブレークポイント

行番号をクリック	行ブレークポイントの切り替え
右クリック -> 条件付き	式が true のときのみ停止
右クリック -> ログポイント	実行を一時停止せずにログ記録
DOM ブレークポイント	サブツリー/属性/削除で停止
XHR/Fetch ブレークポイント	URL が文字列を含むときに停止
イベントリスナーブレークポイント	特定のイベント種別で停止

デバッガーコントロール

F8 / Ctrl+	実行を再開 / 一時停止
F10	次の関数呼び出しをステップオーバー
F11	関数呼び出しにステップイン
Shift+F11	現在の関数からステップアウト
Ctrl+Shift+P -> "never pause"	すべてのブレークポイントを無効化

デバッグパネル

Watch	式の値を監視
Scope	ローカル、クロージャ、グローバル変数
Call Stack	関数呼び出しチェーン
Snippets	再利用可能な JS コードを保存・実行

パフォーマンス

記録

Record ボタン（Ctrl+E）	パフォーマンスの記録を開始/停止
Reload ボタン	ページロードのパフォーマンスを記録
Screenshots チェックボックス	ビジュアルタイムラインをキャプチャ
CPU スロットルのドロップダウン	4x/6x の CPU 低速化をシミュレート
ネットワークスロットル	記録中に低速ネットワークをシミュレート

フレームチャートの読み方

Main トラック	JavaScript の実行（フレームチャート）
Network トラック	ネットワークリクエストのタイムライン
Frames トラック	FPS とフレーム時間
Timings トラック	FCP、LCP、DCL、Load マーカー
黄色のバー	JavaScript（スクリプティング）
紫色のバー	レイアウト / レンダリング
緑色のバー	ペイント / コンポジット

ボトムアップとサマリー

Summary タブ	時間の内訳: スクリプティング、レンダリングなど
Bottom-Up タブ	コストの高い関数を上から表示
Call Tree タブ	ルートから末端への呼び出し階層
Event Log タブ	時系列イベント一覧

アプリケーション

ストレージ

Local Storage	オリジンごとのキー/値ペアを表示・編集
Session Storage	セッションスコープのストレージを表示・編集
IndexedDB	オブジェクトストアとレコードを検査
Cookies	ドメインごとの Cookie を表示・編集・削除
Cache Storage	Service Worker のキャッシュを検査
ストレージをクリア	選択したストレージ種別を一括クリア

Service Workers とマニフェスト

Service Workers	登録状況、ステータス、プッシュ/同期を表示
リロード時に更新	リロードのたびに SW を強制更新
ネットワークにバイパス	SW をスキップしてネットワークに直接アクセス
Manifest	Web アプリマニフェストの詳細を表示
Offline チェックボックス	オフラインモードをシミュレート

Lighthouse

監査の実行

Mode: Navigation	フルページロード監査
Mode: Timespan	一定時間のユーザー操作を監査
Mode: Snapshot	現在のページ状態を監査
Categories	パフォーマンス、アクセシビリティ、ベストプラクティス、SEO
Device	モバイルまたはデスクトップのシミュレーション

主要指標

FCP（First Contentful Paint）	最初のコンテンツが表示されるまでの時間
LCP（Largest Contentful Paint）	最大のビジュアル要素が表示されるまでの時間
TBT（Total Blocking Time）	ロングタスクによるブロック時間の合計
CLS（Cumulative Layout Shift）	視覚的安定性スコア
SI（Speed Index）	コンテンツがどれだけ速く視覚的に表示されるか
INP（Interaction to Next Paint）	ユーザー入力への応答性

ショートカット

DevTools を開く

F12 / Ctrl+Shift+I	DevTools を開く/閉じる
Ctrl+Shift+J	Console パネルを開く
Ctrl+Shift+C	Elements を開いて検査モードに
Ctrl+Shift+M	デバイスツールバーを切り替え（レスポンシブ）

ナビゲーションと検索

Ctrl+Shift+P	コマンドメニュー（任意のアクションを実行）
Ctrl+P	ファイルを開く（ファイルに移動）
Ctrl+Shift+F	すべてのソースを横断検索
Ctrl+G	Sources で行番号に移動
Ctrl+[/ Ctrl+]	パネルを切り替え

Chrome DevTools クイックリファレンス

編集とコンソール

Ctrl+Shift+D	DevTools のドック位置を切り替え
Ctrl+L（コンソール）	コンソール出力をクリア
Shift+Enter（コンソール）	複数行入力
Esc	コンソールドロワーを切り替え
Ctrl+K（コンソール）	コンソールをクリア

モバイルデバッグ

デバイスツールバー

Ctrl+Shift+M	デバイスツールバーの切り替え
デバイスのドロップダウン	プリセットの電話/タブレットサイズ
レスポンシブモード	ビューポートを自由にリサイズ
DPR のドロップダウン	デバイスピクセル比を変更
スロットリング	モバイル CPU とネットワークをシミュレート
メディアクエリを表示	CSS ブレークポイントを可視化

リモートデバッグ（Android）

1. USB デバッグを有効化	デバイスの設定 → 開発者オプション
2. USB で接続	Android デバイスをコンピューターに接続
3. chrome://inspect	デスクトップの Chrome で開く
4. 検査をクリック	モバイルページの DevTools を開く

デスクトップと Android デバイスの両方で Chrome が必要

センサーのオーバーライド

Geolocation	GPS 座標を上書き
Orientation	デバイスの向きをシミュレート
Touch	タッチイベントをシミュレート
Idle detection	アイドル検出 API を上書き

よくあるパターン

ネットワーク問題のデバッグ

```
// コンソールですべての fetch リクエストを監視
const origFetch = window.fetch;
window.fetch = (...args) => {
  console.log('fetch:', args);
  return origFetch(...args);
};
```

パフォーマンスのワークフロー

1. Lighthouse 監査	主要なパフォーマンス問題を特定
2. パフォーマンス記録	フレームチャートでロングタスクを探す
3. Coverage タブ	未使用の CSS/JS を探す（Ctrl+Shift+P -> Coverage）
4. ネットワーク ウォーターフォール	ブロッキングリソースを特定
5. Rendering タブ	ペイント/レイアウトを可視化（Ctrl+Shift+P -> Rendering）

便利なコンソールスニペット

```
// 要素のすべてのイベントリスナーを一覧表示
getEventListeners($0);

// レイアウトシフトを監視
new PerformanceObserver(l => l.getEntries().forEach(
  e => console.log('CLS:', e)
)).observe({ type: 'layout-shift', buffered: true });
```