

JavaScript 開発環境の準備と使い方

JavaScript の実行エンジンは Web ブラウザに搭載されているので、Web ブラウザとエディタがあれば、JavaScript のプログラムを開発することができる。

JavaScript の開発環境として、オフライン環境には、Windows に標準搭載されているテキストエディタの「メモ帳」や VSCode (Visual Studio Code) などがある。一方、オンライン環境には、ユーザ登録不要な jsFiddle やスマートフォンのアプリケーションが開発できる Monaca などがある。

JavaScript は基本的にどのような Web ブラウザでも動作するが、Internet Explorer はクラスベースのプログラムは動作せず、2022 年 6 月でサポート終了となるので、Google Chrome や Microsoft Edge などの Web ブラウザを使用するとよい。

1. JavaScript の有効の設定

JavaScript は Web ブラウザで実行されるプログラミング言語であるため、Web ブラウザの JavaScript の機能を有効しておく必要がある。

(1) Google Chrome の場合

Chrome を起動し、右上の設定「⋮」をクリックし、「プライバシーとセキュリティ」、「サイトの設定」、「JavaScript」の順に選択し、スイッチをオン（青色）にする。

(2) Microsoft Edge の場合

Edge を起動し、右上の設定「⋮」をクリックし、「Cookie とサイトのアクセス許可」、「JavaScript」の順に選択し、スイッチをオン（青色）にする。

2. メモ帳

Windows に標準搭載されている「メモ帳」などのテキストエディタを利用して、HTML 内の<script>タグと</script>タグの間に組み込んだ JavaScript のプログラムを入力する。ファイルの種類を「すべてのファイル」、拡張子を「.html」、文字コードを「UTF-8」にして保存する。

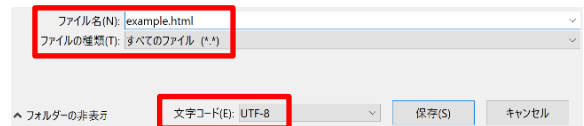


図1 メモ帳（ファイル保存時）

保存したファイルをダブルクリックすると、Web ブラウザが開き、プログラムが実行される。

ダブルクリックしても Web ブラウザが開かない、もしくは別のブラウザで開きたい場合は、「ファイル名.html」のアイコンを右クリックし、「プログラムから開く」をクリックし、表示されるメニューから開きたい Web ブラウザを選択する。

なお、Internet Explorer で開く場合は、「この Web ページはスクリプトや ActiveX コントロールを実行しないように制限されています。」と表示される場合がある。このような場合は「ブロックされているコンテンツを許可」をクリックする。

メモ帳では Tab キーによるインデント（字下げ）の幅は半角8文字分に固定されており変更することができない。Tab キーは押す回数が少ない反面、環境によって見え方が異なることからスペースキーによるインデントが推奨されているため、コード内のインデントは半角スペースキーを押して設定するようになる。

3. VSCode

前述の Windows に付属のテキストエディタであるメモ帳だけでもプログラムの開発ができるが、Visual Studio Code (VSCode) を使用すると、より効率的な開発ができる。

(1) VSCode のインストール

以下の URL からダウンロードしてインストールする。

<https://code.visualstudio.com/>

インストール後のデフォルトのインデントと文字エンコードは、それぞれ「半角スペース 4 つ」と「UTF-8」になっているが、VSCoDe の画面下部でこれらを変更することができる。

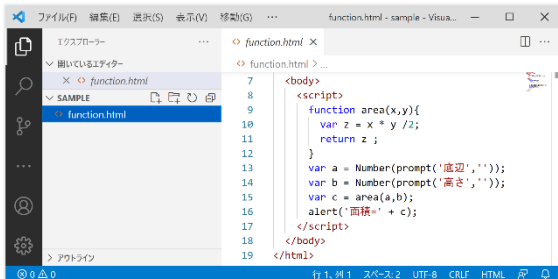


図 2 Visual Studio Code

(2) 日本語表示の拡張機能のインストール

上部のメニューから「View」、「Extensions」の順に選択して表示される検索ボックスに「japanese」を入力すると、「Microsoft」の「Japanese Language Pack for Visual Studio Code」が表示されるので、その中の右下にある「Install」をクリックする。「Restart」をクリックして再起動すると、メニューなどが日本語表示される。

(3) VSCoDe の簡単な使い方

① フォルダを開く

ファイルを保存するフォルダを選択するために、上部のメニューから「表示」、「エクスプローラー」、「ファイル」、「フォルダを開く…」の順に選択すると、「フォルダを開く」というウィンドウが開く。

ファイルを保存するためのフォルダを選択してから「フォルダの選択」をクリックすると、左に選択したフォルダが表示される。新規フォルダを作成するには「新しいフォルダ」をクリックする。

② 新規ファイルの作成

上記 ① の操作後に、「ファイル」、「新規ファイル」の順に選択すると、上部に「Untitled-1」という仮のファイル名でタブが表示される。

続けて「ファイル」、「名前を付けて保存…」の順に選択し、「ファイル名.html」を入力し、「保存」をクリックする。

③ 既存ファイルのオープン

上記 ① の操作後に、上部のメニューから「ファイル」、「ファイルを開く…」の順に選択し、ファイル(フ

イル名.html)を選択してから、「開く」をクリックする。

④ プログラムの入力

HTML 内の<script>タグと</script>タグの間に組み込んだ JavaScript のプログラムを右部のエディタエリアに入力する。

⑤ ファイルの上書き保存

「ファイル」、「保存」の順に選択する。

⑥ プログラムの実行

・操作方法例 1

保存した「ファイル名.html」のアイコンをダブルクリックすると、Web ブラウザが開き、プログラムが実行される。

ダブルクリックしても Web ブラウザが開かない、もしくは別のブラウザで開きたい場合は、「ファイル名.html」のアイコンを右クリックし、「プログラムから開く」をクリックし、表示されるメニューから開きたい Web ブラウザを選択する。

・操作方法例 2

VSCoDe の左部または上部に表示されている「ファイル名.html」の上で右クリックし、「パスのコピー」をクリックする。Web ブラウザの URL 入力欄にコピーしたものを貼り付け、「Enter」を押すことで、プログラムが実行され、Web ブラウザに実行結果が表示される。

・操作方法例 3

「open in browser」という拡張機能を利用する方法がある。

上部のメニューから「表示」、「拡張機能」を選択して表示される検索ボックスに「open in browser」と入力すると、「TechER」の「open in browser」が表示されるので、その中の右下にある「インストール」をクリックすると、編集している HTML ファイルをすぐに Web ブラウザで開くことができる。

Alt+B キーを押すと現在開いているファイルを Web ブラウザで表示することができる。

また Shift+Alt+B キーを押すと、Web ブラウザの種類を選択することもできる。

4. JSFiddle

JSFiddle (ジェイエスフィドル) は、Web ブラウザ上で HTML, CSS, JavaScript のコードを記述し、その動作を確認できるオンラインの Web サービスである。無料でユーザ登録は不要であるが、登録しても無料で使え、自分が作成したコードの保存や読み出し、削除などができる。

(1) コードの作成

以下の URL に接続する。

<https://jsfiddle.net/>

JSFiddle は、図 3 のように画面を 4 つに分割して、左上に「HTML」、右上に「CSS」、左下に「JavaScript」、右下に「Result」の各エリアを同時に表示できるようになっている。

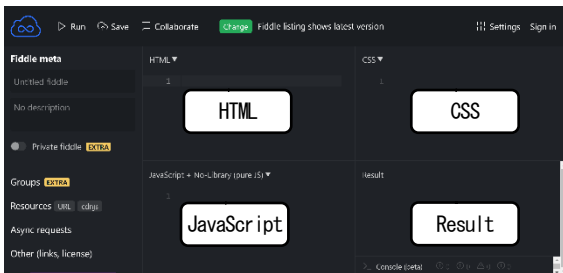


図 3 JSFiddle

「HTML」エリアには HTML の `<body>` タグの中身を記述する。`<html>` や `<body>`, `<head>` タグは、背後で挿入されるため、記述は不要である。

「CSS」エリアには CSS を記述する。スタイルシートを設定するための `<style>` タグは、背後で挿入されるので記述は不要である。

「JavaScript」エリアには JavaScript コードを記述する。JavaScript コードを設定するための `<script>` タグは、背後で挿入されるので記述は不要である。

なお、「CSS」と「JavaScript」エリアに記述せずに、「HTML」エリアに `<html>` や `<body>`, `<head>`, `<style>`, `<script>` タグを使って HTML, CSS, JavaScript のすべてのコードを記述してもよい。

(2) 実行

上部にあるメニューの「Run」ボタンを押すとコードの実行結果が「Result」エリアに表示される。

(3) コードの保存

メニューの「Save」をクリックすると、コード

がクラウド上に保存される。保存すると Web ブラウザのアドレスバーに URL が発行される。

ユーザ登録しない場合は、保存の際に発行される URL をその都度記録しておき、この URL を Web ブラウザのアドレスバーに入力することで保存したコードの読み出しや他人と共有ができるが、コードの削除はできない。

ユーザ登録してログインすると、保存したコードの読み出しや削除ができるため、URL を記録しておく手間を省くことができる。

「save」を繰り返すと URL 末尾の数字が増えていくが、これはバージョンを表している。


また、メニューの「Fork」をクリックすると、別の URL が発行される。

(4) ユーザ登録

以下の URL に接続する。

<https://jsfiddle.net/>

右上の「sign in」をクリックし、下部の「Sign up」を入力する。「Username」、「E-mailAddress」、「Password」を入力し、「Create an account」をクリックする。なお、ユーザ登録が完了しても登録完了のメールは通知されない。

右上のアカウントのアイコン  , 「Logout」の順にクリックしてログアウトする。


(5) ログイン

以下の URL に接続する。


<https://jsfiddle.net/>

右上の「sign in」をクリックし、ユーザ名とパスワードを入力し、「Log in」をクリックする。


(6) ログアウト


右上のアカウントのアイコン  , 「Logout」の順にクリックする。

(7) 保存したコードの読み出し

ログイン後、右上のアカウントのアイコン  , 「Your fiddles」の順にクリックすると、保存したコードの一覧が表示されるので、読み出したいコードの下に表示される「英数字の番号」をクリックする。

(8) 保存したコードの削除

ログイン後、右上のアカウントのアイコン  , 「Your fiddles」の順にクリックすると、

保存したコードの一覧が表示されるので、削除したいコードの下に表示される歯車のアイコン  , 「Delete fiddle」, 「OK」の順にクリックする。

(9) 注意点

JSFiddleは保存時に発行されるURLを伝えるだけで、他人にコードを共有することができる。

しかし、URLを共有すると誰でもアクセス可能になるため、個人情報や著作権を含むコードを入力しないようにする必要がある。

5. Monaca

(1) Monaca とは

Monaca は、HTML5 や CSS, JavaScript を用いて iOS や Android 向けのスマートフォンやタブレット用のアプリケーションなどを Web ブラウザ上で開発するためのプラットフォームで、アシアル(株)が提供している。

作成したアプリケーションを生徒のスマートフォンなどですぐに動作させることができるので、楽しくプログラミングを学ぶことができる。

また、JavaScript から Cordova (コルドバ) と呼ばれるプラグイン (拡張機能) を使うことで、スマートフォンに内蔵されている加速度センサや方位センサ (コンパス), カメラなどの装置を利用した、動きのある魅力的なアプリケーションを比較的容易に作成することもできる。Monaca には、通常版 Monaca のほか、教育版 Monaca (Monaca Education) などのサービスがあり、それぞれ、有料プランや無料プラン (フリープラン) がある。

なお、フリープランはプロジェクト数 3、ストレージ容量 50MB、保存期間 60 日などの機能や利用制限があるため、フリープランで試してみて、自分にあったサービスであれば有料のプランに移行するとよい。ここでは Monaca Education のフリープランの使い方の概要について解説する。

(2) ユーザ登録の方法

以下の URL に接続する。

<https://edu.monaca.io/>


右上の「アカウントの作成」をクリックし、「メールアドレス」と「パスワード」の入力し、「ア

カウント新規作成」をクリックすると仮登録の状態になる。登録したメールアドレスに仮登録のメールが送られてくるので、メール上の「本登録はこちら」をクリックする。

「アクティベーションコードを使う」の部分をクリックすると、「Free プラン」のメニューが表示されるので「Free プラン」を選択する。

名前を入力したのち、「次に進む」, 「OK」, 「ダッシュボードに進む」, 「同意する」の順にクリックするとダッシュボードの画面が表示される。

ここで終了する場合はログアウトする。

ログアウトの方法は右上のアカウントのアイコン  をクリックし、「ログアウト」, 「OK」の順にクリックする。

(3) ログインの方法

以下の URL に接続する。

<https://edu.monaca.io/>

右上の「ログイン」をクリックし、「メールアドレス」と「パスワード」の入力し、「ログイン」をクリックすると、ダッシュボードが表示される。

(4) プロジェクトの作成

ダッシュボードの画面の中で、「新しいプロジェクトを作る」, 「最小限のテンプレート」の順にクリックする。

「プロジェクト名」の部分にある「最小限のプロジェクト」を削除してプロジェクト名を入力し、「説明」の部分に説明文を入力し「作成」をクリックする。作成したプロジェクト名のアイコンをクリックし、「クラウド IDE で開く」すると、図 4 のような Monaca クラウド IDE が表示される。

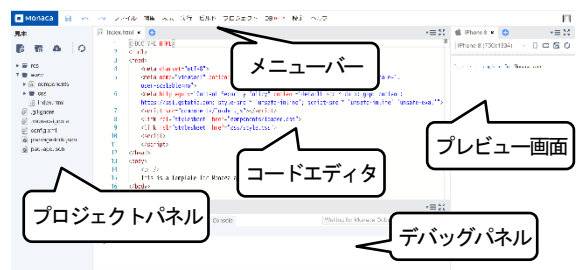


図 4 Monaca クラウド IDE

(5) プログラムの作成

図 4 の Monaca クラウド IDE 画面で JavaScript


のコードやHTMLなどを編集する。

画面中央上部のコードエディタにひな形のHTMLが表示されている。

`<body>`と`</body>`の間にある「This is a template for Monaca app.」は、右のプレビュー画面に表示されている。ここでは、この部分を変更してみる。

「This is a template for Monaca app.」を例えば、「はじめてのプログラム」に変更し、メニューバーの「ファイル」、「保存」の順にクリックすると、プレビュー画面に「はじめてのプログラム」が表示される。なお、HTMLを変更すると「index.html」が「index.html*」になり、保存すると「index.html」に戻る。JavaScriptのコードは、コードエディタに表示されている`<script>`と`</script>`の間にJavaScriptのコードを入力する。


例えば「alert('こんにちは');」を入力し、メニューバーの「ファイル」、「保存」の順にクリックすると、実行結果が表示される。

なお、`<script>`と`</script>`は、`<head>`と`</head>`の間に置かれているが、`<body>`と`</body>`の間に置いてもよい。また、再実行する場合は右の更新ボタン  をクリックする。

(6) プロジェクトファイルのエクスポート

Monaca クラウドIDE画面で、「プロジェクト」、「エクスポート」、「OK」の順にクリックし、保存フォルダを選択し、ファイル名を入力してから「OK」をクリックするとZIP形式のファイルがPCにダウンロードされる。

(7) IDE からダッシュボードに戻る

右上の「アカウントのアイコン」  , 「ダッシュボードに戻る」の順にクリックする。

(8) プロジェクトファイルのインポート


ダッシュボード画面で、「インポート」、「ZIPファイル」、「Choose a file」をクリックし、インポートするZIPファイルを選択してから「開く」をクリックする。「プロジェクト名」と「説明」を入力し、「プロジェクトのインポート」をクリックすると、プロジェクトファイルがアップロードされる。


(9) プロジェクトの削除

ダッシュボード画面で、削除するプロジェクトの左横のチェックボックスをクリックしてチェッ

クを入れ、「削除」をクリックする。

(10) ログアウト

Monaca クラウドIDE からログアウトする場合は、右上の「アカウントのアイコン」  , ログアウト」、「OK」の順にクリックする。

ダッシュボードからログアウトする場合は、右上の「アカウントのアイコン」  , 「ログアウト」、「OK」の順にクリックする。

(11) Monaca for Study のインストール

Monaca for Study と呼ばれるデバッガアプリを使うとソースコードを保存するだけで、スマホ上でアプリの動作を確認できる。

動作確認のために、ソースコードを元に実行可能ファイルを作成するためのビルドという操作をしたり、ビルドしたファイルをスマートフォンにインストールしたりする必要がない。

教育版ではMonaca for Study をスマートフォンにインストールする。通常版とはアプリ名が異なるので注意したい。スマートフォンがiPhoneの場合はApp Storeで、Androidの場合はGoogle Playで「Monaca for Study」というキーワードで検索してそれぞれインストールする。

(12) Monaca for Study での実行

インストール後にスマートフォンでMonaca for Study を起動し、Monaca に登録したときに使用したメールアドレスとパスワードを使ってログインする。ログインすると開発中のプロジェクト一覧が表示されるので、動作確認したいアプリケーションの名前をタップすると、実行結果が表示される。再実行する場合は、右下に表示される丸いボタンをタップすると、メニューが表示されるので、丸い矢印の形のボタンをタップする。

終了する場合は、スマートフォンの「戻る」ボタン、左下の「アカウントのアイコン」、「ログアウト」の順にタップする。

なお、動作確認が完了したアプリケーションをApp Store やGoogle Play ストアで配布するには、ビルドという作業や、アプリの提出に必要な手続き、さまざまな処理や操作をする必要があるが、「情報」の授業の本質から外れるため、Monaca for Study での動作確認だけに留めるとよい。

6. Chrome デベロッパーツールによるデバッグ

デバッグは、プログラムの欠陥 (バグ) を発見・修正するデバッグ作業を支援するソフトウェアである。デバッグを使うと、ブレークポイント (設定した場所で処理を一時停止させる機能) やステップ実行 (プログラムを1行ずつ実行する機能) などを使って、変数の値を調べたり関数の呼び出し履歴をみたりできるので、問題箇所を発見することができる。Web ブラウザの Google Chrome には JavaScript のデバッグを行うデベロッパーツールが用意されている。

(1) デベロッパーツールの起動

JavaScript コードが含まれている HTML ファイルを Google Chrome で開き、「F12」キーを押すか、画面の上で右クリックして「検証」をクリックすると、図5のように表示される。

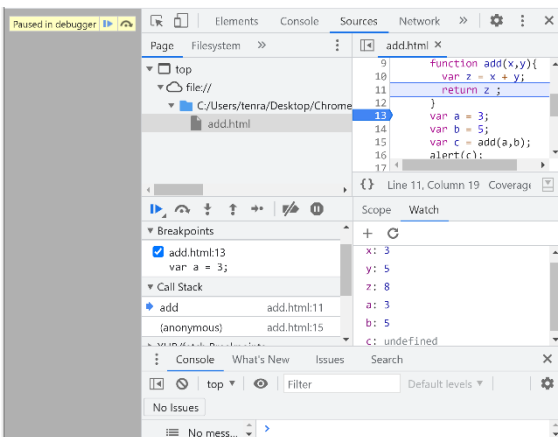


図5 Google Chrome デベロッパーツール

上部の「Sources」をクリックすると、該当のHTMLファイルが表示される。表示されない場合は、ファイル名の部分をクリックする。

(2) 確認する変数の設定

実行途中で、変数の値を確認することができる。

画面中央右のメニューから「watch」,「+」の順にクリックし、変数名を入力し、クリックする。

クリックしないで Enter を押すと、正しく入力できない場合があるので変数名入力後はクリックする。変数とその値は「変数名:値」として表示される。確認する変数は複数設定できる。

変数の削除は、設定した変数の上にマウスポイ

ンタを重ねたときに表示される「x」をクリックする。

(3) ブレークポイントの設定

プログラムの実行を一時停止させるブレークポイントは、HTML の<script>〜</script>内にある JavaScript プログラムの行番号をクリックする。クリックすると行番号が青色に変わり、ブレークポイントが設定される。もう一度クリックすると、設定が解除される。

(4) ステップ実行で使用するボタン

ブレークポイントで停止したプログラムは、図6のボタンで続きを実行できる。図6のボタンは後述の「(5)ステップ実行」でブレークポイントでプログラムが停止したときに表示される。



図6 ステップ実行で使用するボタン

・リジューム実行

リジューム実行は、ブレークポイントで停止した残りの処理をすべて実行する。

・ステップイン

ステップインは1行ごとに実行する。すなわち、現在の行を実行し、次の行へ移動して停止する。関数内でも1行ずつ実行する。

・ステップオーバー

ステップオーバーはステップインと同様に1行ごとに実行するが、関数があった場合はそれを1行とみなして実行する。すなわち、関数呼び出しがあると、その関数の内部の処理をすべて実行してから戻ったところで停止する。

・ステップアウト


ステップアウトは関数内で実行中にクリックすると、その関数の内部の処理をすべて実行してから戻ったところで停止する。

(5) ステップ実行

ここでは、図7のプログラムをステップ実行して、使われている各変数の値の変化を調べる方法について説明する。

```
1 <!DOCTYPE html>
2 <html lang = "ja">
3   <head>
4     <meta charset = "utf-8">
5     <title>足し算</title>
6   </head>
7   <body>
8     <script>
9       function add(x,y){
10        var z = x + y;
11        return z ;
12      }
13      var a = 3;
14      var b = 5;
15      var c = add(a,b);
16      alert(c);
17    </script>
18  </body>
19 </html>
```

図7 デバッグするサンプルプログラム

13行目の「var a = 3;」にブレークポイントを設定し、Chrome の上部のアドレスバーの横にあるリロードボタン  をクリックしてプログラムを実行させる。ブレークポイントでプログラムの実行が一時停止され、図8のように13行目の「var a = 3;」は水色で網掛けされる。

この網掛けのコードはまだ実行されていないことを示している。画面下部の「watch」で設定した変数の値が確認できる。網掛けの13行目はまだ実行されていないので、グローバル変数 a は「undefined : 未定義」となっている。

他のグローバル変数 b, c も「undefined」となっている。また、ローカル変数 x, y, z は「not available: 利用不可」となっている。

なお、コード中の変数の上にマウスポインタを重ねることで変数の値を確認することもできる。

このあと図6の「ステップイン」ボタンをクリックしてステップ実行でデバッグを進めていく。

図8の状態から「ステップイン」ボタンをクリックすると、図9のように14行目の「var b = 3;」が網掛けされ、「a: undefined」から「a: 3」に変化し、aに3が代入されたことが確認できる。

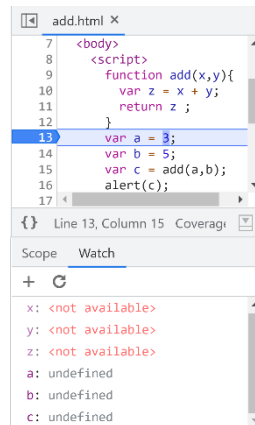


図8 図7の13行目

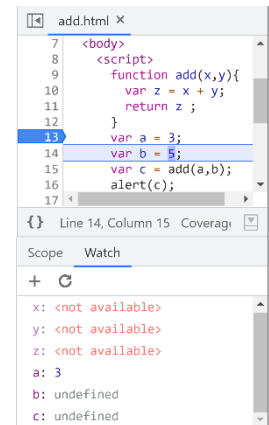


図9 図7の14行目

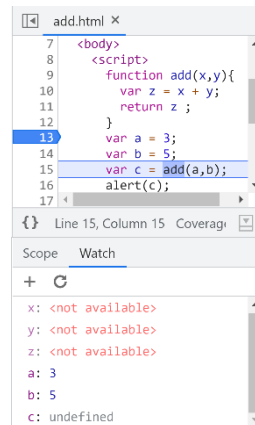


図10 図7の15行目

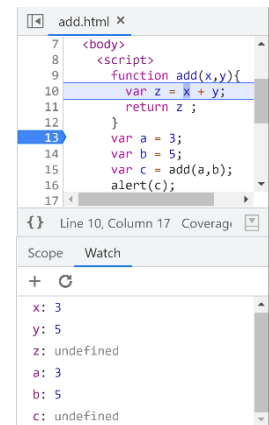


図11 図7の10行目

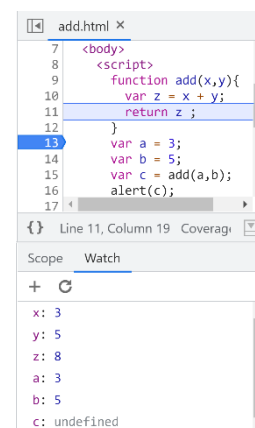


図12 図7の11行目

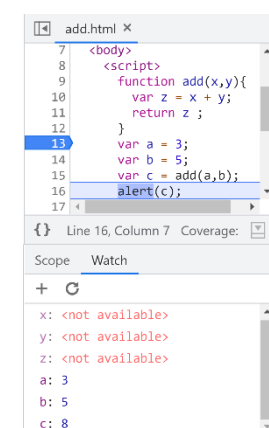


図13 図7の16行目

さらに「ステップイン」をクリックすると、図10のように15行目の「var c = add(a, b);」が網掛けされ、「b: undefined」から「b: 5」に変化する。

続けて「ステップイン」をクリックすると、図11のように関数の内部に入って10行目が網掛け

される。関数の内部に入ると、ローカル変数の x, y, z に値が設定され、図 12 の「return z;」で関数から抜け出し、図 13 の状態になる。

このように「step」ボタンをクリックしていくと、図 8→図 9→図 10→図 11→図 12→図 13 のように網掛けが移動していく。

表 1 の左端列の番号は「step」ボタンをクリックするごとに、網掛けの部分の行番号が 13→14→15→10→11→16 の順に変化することを表している。

2列目と3列目は、図 8～図 13 の下部に表示されるローカル変数とグローバル変数の値をそれぞれ表している。16行目が網掛けになった状態で「ステップイン」ボタンをクリックすると「alert(c);」が実行され、alert 画面に計算結果の 8 が表示される。最初の状態に戻るには、図 6 の「リジューム実行」をクリックする。

表 1 表示される変数の値

網掛けの表示順 (図 7 の行番号)	ローカル変数			グローバル変数		
	x	y	z	a	b	c
13						
14				3		
15				3	5	
10	3	5		3	5	
11	3	5	8	3	5	
16				3	5	8

(6) エラー表示

コードに文法上のエラーがあると、エラーメッセージが表示される。例えば、図 7 の 16 行目を「arert(a);」に変更して実行すると、図 14 のエラーメッセージが画面下部の「console」に表示される。

また、コードの 16 行目に赤色の「×」が表示され、それをクリックすると、図 15 のエラーメッセージが表示される。

at add.html:16

図 14 console 部分のエラーメッセージ

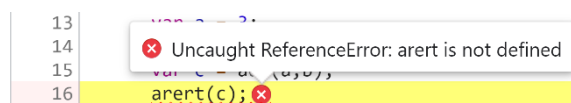


図 15 コード部分のエラーメッセージ