HTML5 Professional Certification

HTML5プロフェッショナル認定 レベル2 技術解説無料セミナー

2023/12/9 開催

主題	2.3.1 Canvas(2D)
副題	Canvas API の主要なメソッドを学ぶ

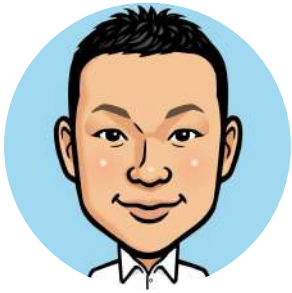
本日の講師

天川村立天川小中学校
ICT教育主任 野口 浩幹

LPI-JAPAN

講師の紹介

- 講師：野口 浩幹（Hiroki Noguchi）
  - 天川村立天川小中学校 英語科教諭、ICT 教育主任
  - HTML5 Professional Level 2 （令和4年8月取得）



- Tech 関連の活動
  - Google 開発者グループ (GDG) Nara　運営（2020年～）

GDG Nara

本日の目標

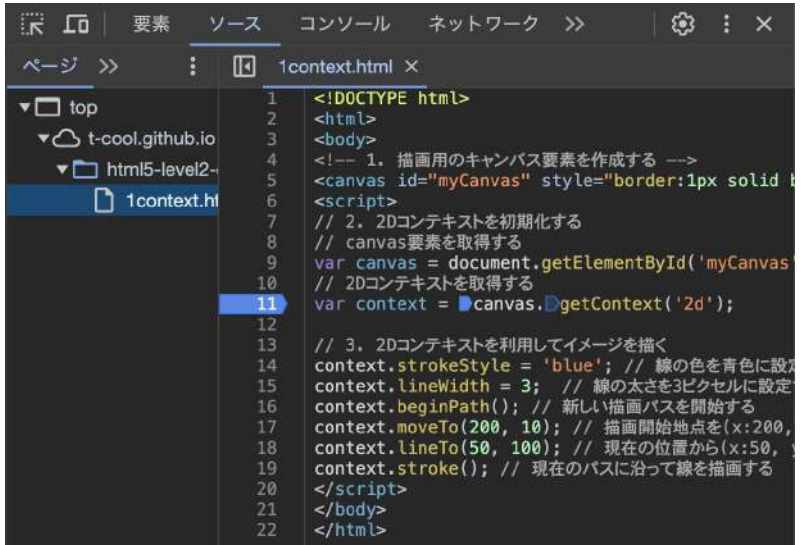
1. HTML5 プロフェッショナル認定試験レベル2の特徴を知る

2. Canvas(2D) の主要なメソッドを知る
   （コードを読む）

3. デベロッパーツールの使い方を知る

Level.2

```
context.strokeStyle = 'blue'; // 線の色を青色に設定する
context.lineWidth = 3;  // 線の太さを3ピクセルに設定する
context.beginPath(); // 新しい描画パスを開始する
context.moveTo(200, 10); // 描画開始地点を(x:200, y:10)に設定する
context.lineTo(50, 100); // 現在の位置から(x:50, y:100)まで線を引く
context.stroke(); // 現在のパスに沿って線を描画する
```



本日の予定

1. HTML5 プロフェッショナル認定試験について
   - レベル1とレベル2の違い
   - 資格取得の意義

2. 2.3.1 Canvas(2D) 解説
   - 6つの項目の技術解説（コード解説）
     （コンテキスト、基本図形描画、テキスト描画、
     変形、エフェクト、イメージデータ）

```
context.strokeStyle = 'blue'; // 線の
context.lineWidth = 3;  // 線の太さを3
context.beginPath(); // 新しい描画パスを
context.moveTo(200, 10); // 描画開始地
context.lineTo(50, 100); // 現在の位置
context.stroke(); // 現在のパスに沿って
```

3. まとめ
   - 確認クイズ
   - 質疑応答
   - 学習リソースの紹介

HTML5プロフェッショナル認定試験とは？

■ HTML5プロフェッショナル認定試験とは

HTML5プロフェッショナル認定とは、HTML5、CSS3、JavaScriptなど最新のマークアップに関する技術力と知識を、公平かつ厳正に、中立的な立場で認定する認定制度です。

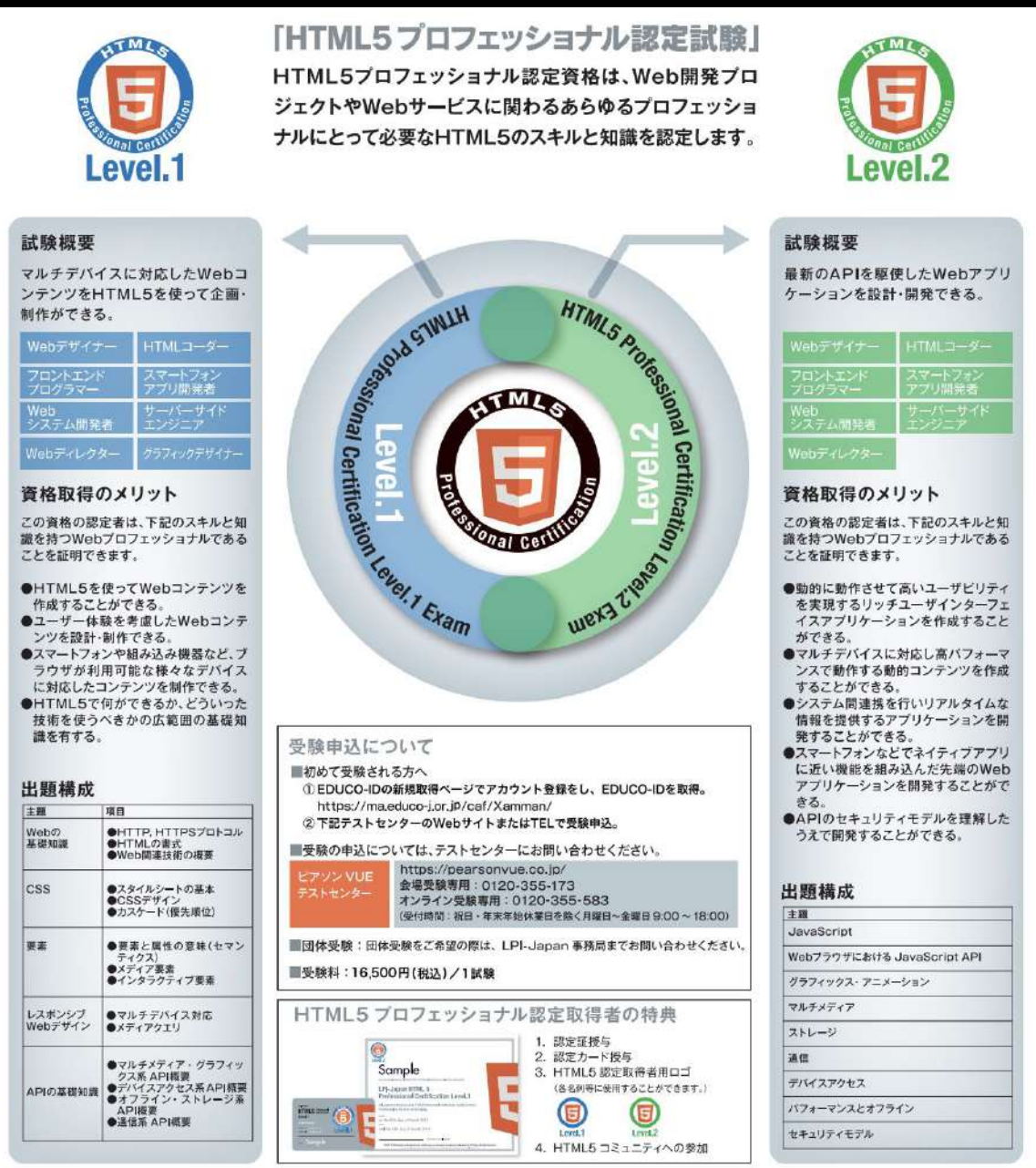
✔レベル1では HTML と CSS

マルチデバイスに対応したWebコンテンツ制作の基礎の実力を測る

✔レベル2では JavaScript

システム間連携や最新のマルチメディア技術に対応したWebアプリケーションや動的Webコンテンツの開発・設計の能力を認定する

（参照：公式サイト｜試験概要）



HTML5はHTML Standardへ呼称変更

～ HTML5プロフェッショナル認定試験は今後も有効 ～

2021年からHTML5は「HTML Standard」と呼称が変わりました。HTML Living Standardと呼ばれることもあります。しかし、ご安心を。これまで通り、HTML StandardがWebを支える標準として進化を続けます。またHTML5プロフェッショナル認定試験についても、現状、出題範囲に影響を与える部分はほとんどなく、認定の価値に変わりはありません。今後もWebエンジニアのスキル認定として有効です。

（ 引用：https://html5exam.jp/measures/column_01.html）

HTML5プロフェッショナル認定試験 レベル2の出題範囲について

2.1 JavaScript
- 2.1.1 JavaScript文法(重要度:10)
- 2.1.2 ES6(ECMAScript2015)以降の新機能(重要度:8)

2.2 WebブラウザにおけるJavaScript API
- 2.2.1 イベント(重要度:8)
- 2.2.2 ドキュメントオブジェクト／DOM(重要度:6)
- 2.2.3 ウィンドウオブジェクト(重要度:8)
- 2.2.4 Selectors API(重要度:7)
- 2.2.5 History API(重要度:7)
- 2.2.6 テスト・デバッグ(重要度:6)

2.3 グラフィックス・アニメーション
- **2.3.1 Canvas(2D)(重要度:8)**
- 2.3.2 SVG(重要度:2)
- 2.3.3 Timing control for script-based animations(重要度:2)

2.4 マルチメディア
- 2.4.1 メディア要素のAPI(重要度:5)

2.5 ストレージ
- 2.5.1 Web Storage(重要度:7)
- 2.5.2 Indexed Database API(重要度:5)
- 2.5.3 File API(重要度:5)
- 2.5.4 バイナリーデータ(重要度:4)

2.6 通信
- 2.6.1 Web Socket(重要度:5)
- 2.6.2 XMLHttpRequest(重要度:5)
- 2.6.3 Server-Sent Events(重要度:1)

2.7 デバイスアクセス
- 2.7.1 Geolocation API(重要度:5)
- 2.7.2 DeviceOrientation Event(重要度:1)

2.8 パフォーマンスとオフライン
- 2.8.1 Web Wokers(重要度:5)
- 2.8.2 High Resolution Time(重要度:2)
- 2.8.3 オフラインアプリケーションAPI(重要度:3)
- 2.8.4 Page Visibility(重要度:2)
- 2.8.5 Navigation Timing(重要度:1)

2.9 セキュリティモデル
- 2.9.1 クロスオリジン制約とCORS(重要度:4)
- 2.9.2 セキュリティモデルとSSLの関係(重要度:4)

(引用:https://html5exam.jp/outline/objectives_lv2.html)

レベル2 取得の魅力

- JavaScript を用いたブラウザの多様な機能（API）の使用方法について広範囲にわたり学習できる。
- ブラウザベースで開発可能なアプリケーションの範囲が広がる。

File API

ファイルの読み込み

DOM API

HTMLやCSSの操作

Geolocation API

位置情報の取得

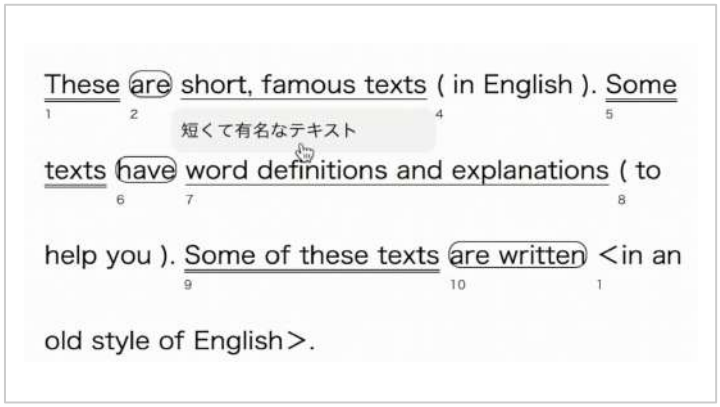
High Resolution Time API

高い精度の時間計測

Canvas API

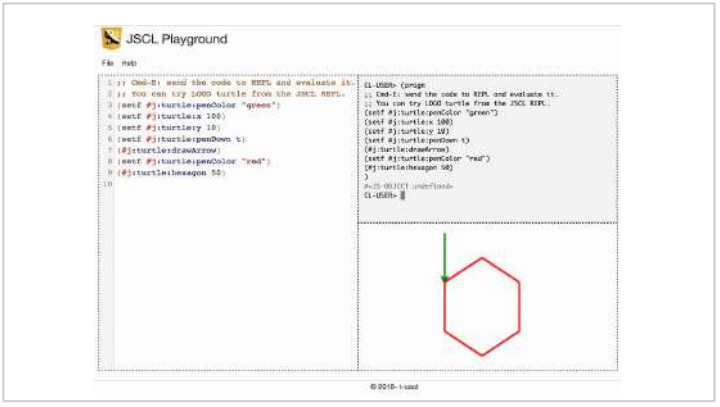
グラフィックの描写

アプリケーションの例



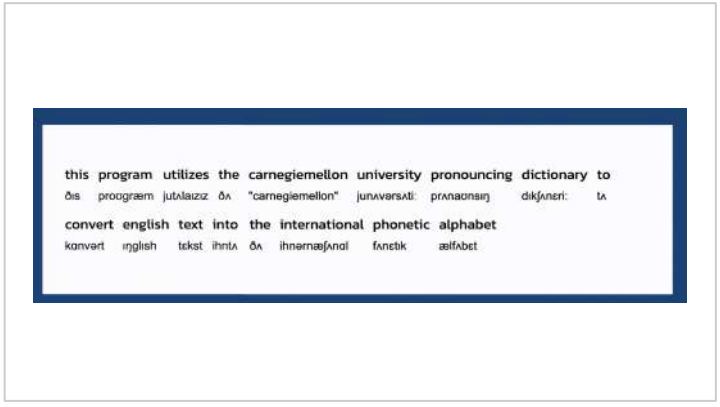
[marken]
- 英文に記号と解説を付与するアプリ
- DOM

[scan-qrcodes]
- 複数の QR コードを識別するアプリ
- Canvas, MediaDevices, DOM, File

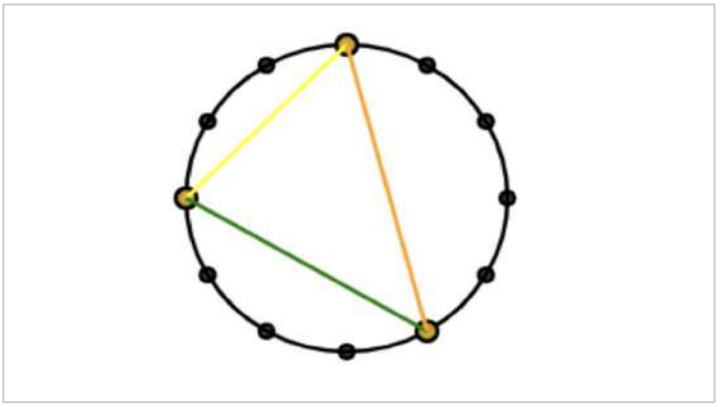


[JSCL Playgroud]
- Lisp で図形を描くアプリ
- Canvas, DOM

[nantango]
- 未知語からプリントを生成するアプリ
- DOM



[phonetic-symbols-on-English]
- 英文に発音記号を付与するアプリ
- DOM, XMLHttpRequest



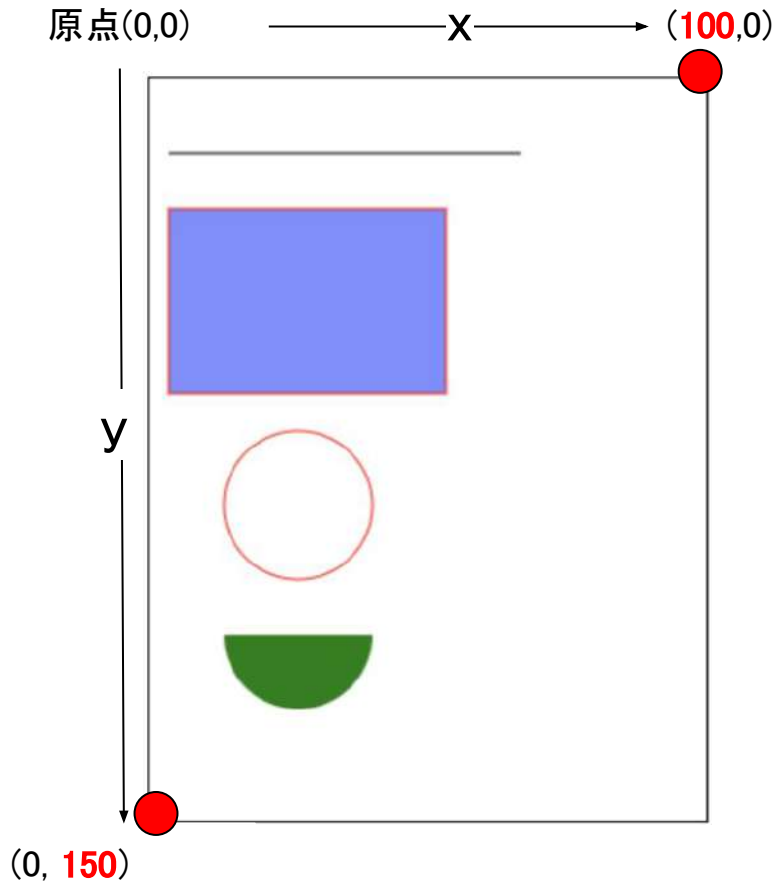
[triangle]
- 図形の学習アプリ
- Canvas, DOM

Canvas (2D)とは？

Canvas(2D)とは？

Canvas(2D) とは？

- Canvas(2D)は、HTML5 から導入されたグラフィック描画機能。
- グラフィックの描画やイベント処理が可能であり、アート制作やゲーム開発等、インタラクティブなグラフィックコンテンツの開発が可能。
- HTML内で <canvas> タグを使用することで描画領域を確保し、JavaScript を用いて描画操作を実行できる。
- 特定の座標にピクセル単位での描画が可能。
- デフォルトでは、キャンバスの左上が原点(0, 0)となり、x軸は左右に、y軸は上下に座標が展開される。



2.3.1 Canvas(2D) の試験範囲

2.3.1 Canvas(2D) の試験範囲

1. **コンテキスト**

   2Dコンテキストの概要と描画状態の遷移、描画状態の保存と復元する方法

   クリッピング領域を指定し描画範囲を制限する方法

2. **基本図形描画**

   線、矩形、曲線描画、Canvasの塗りつぶし

3. **テキスト描画**

   テキスト幅の算定、塗りつぶし描画、輪郭描画、フォントの設定

4. **変形**

   拡大、回転、移動、Canvasの拡大・縮小、回転、移動

5. **エフェクト**

   Canvasへの透明度指定、Canvas上へ図形などを合成

6. **イメージデータ**

   画像の描画

1. コンテキストから
6. イメージデータまで
コードを交えて、
順に要点を説明していきます。

（参照：https://html5exam.jp/outline/objectives_lv2.html ）

2.3.1 Canvas(2D) 解説

「1. コンテキスト」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ コンテキスト
2. 内容
   - ❑ 2Dコンテキストについて
3. ポイント
   - ❑ コンテキストは、2次元の描画を扱う 2Dコンテキストと、
     3次元の描画を中心に扱う webglコンテキストがある。(2Dも描画可)
   - ❑ 試験範囲は 2Dコンテキストのみ。
   - ❑ 2Dコンテキスト(2次元描画コンテキスト)を取得するには、getContext("2d") メソッドを使う。

2Dコンテキストとは？

ここで1つ、たとえ話を。

絵画キャンバスに絵を描く3ステップ

① 描画の場所を確保する

・美術室を予約する

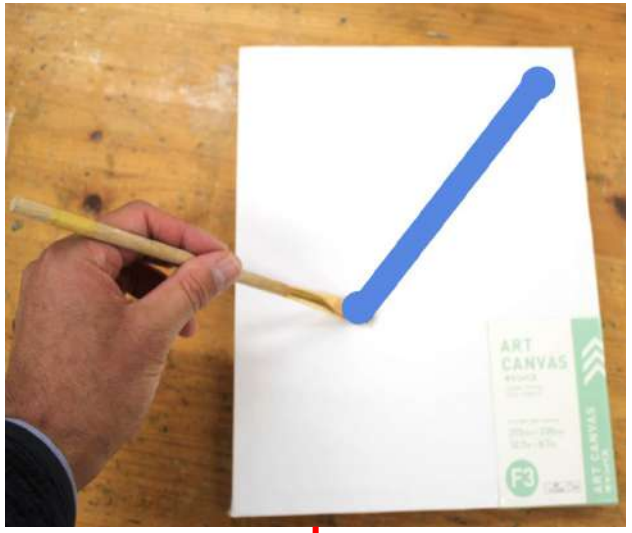
② 描画の道具を用意する

・絵画キャンバス、ブラシ、
絵の具等を準備する

2dコンテキスト

③ イメージを描く

・ブラシの太さや色を決める
・キャンバスに線を引く



**2dコンテキストの道具
(メソッド)で描画を進める**

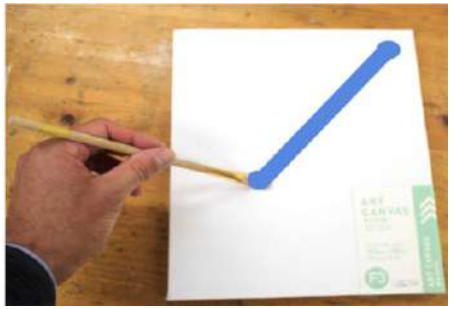
「絵画キャンパス」と「2Dコンテキスト」

① 描画の場所を確保する

・美術室を予約する

② 描画の道具を用意する

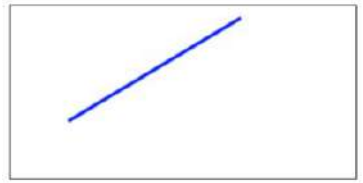
・キャンパス、ブラシ、
絵の具等を準備する



③ イメージを描く

・ブラシの太さや色を決める
・線を描く

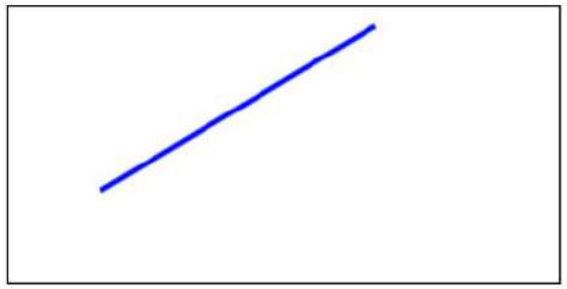
```
<html>
<body>
①
<!--  1. 描画の場所を確保する-->
<!--  canvas 要素を書く--!>
<canvas id='myCanvas'></canvas>
<script>
②
// 2. 2Dコンテキストを初期化する
// canvas要素を取得する
var canvas = document.getElementById('myCanvas');
// 2Dコンテキストを取得する
var context = canvas.getContext('2d');
③
// 3. イメージを描く
context.strokeStyle = 'blue'; // ブラシの色を青色にする
context.lineWidth = 3;  // ブラシの太さを3にする
context.beginPath(); // 描写を開始する
context.moveTo(200, 10); // 開始地点を設定する
context.lineTo(50, 100); // 線を引く
context.stroke(); // 線を描画する
</script>
<body>
</html>
```



コードの例と解説「1. コンテキスト（2Dコンテキスト）」

https://t-cool.github.io/html5-level2-canvas/1context.html

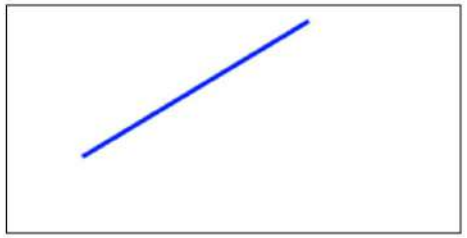
コードの例と解説「1. コンテキスト（2Dコンテキスト）」



```
<!DOCTYPE html>
<html>
<body>
<!-- 1. 描画用のキャンバス要素を作成する -->
<canvas id="myCanvas" style="border:1px solid black;"></canvas>
<script>
// 2. 2Dコンテキストを初期化する
// canvas要素を取得する

// 始点を(x:200, y:10)にする
context.moveTo(200, 10);

// 終点を(x:50, y:100)にする
context.lineTo(50, 100);

// 線を描画する
context.stroke();

</script>
</body>
```

コードの解説は解説動画をご覧ください。

2Dコンテキストの実体（参考までに・・・）



2Dコンテキストの実体は・・・？

- ❑ オブジェクト名.constructor とすると、どの関数からオブジェクトが生成されたかが分かる。

  context.constructor;

  => ƒ CanvasRenderingContext2D()

- ❑ このことから、2Dコンテキストは、CanvasRenderingContext2D から生成されたオブジェクトであることが分かる。

- ❑ デベロッパーツールのコンソールにオブジェクト名（変数名）を打ち込むと、CanvasRenderingContext2D オブジェクトの属性がわかる。

2.3.1 Canvas(2D) 解説

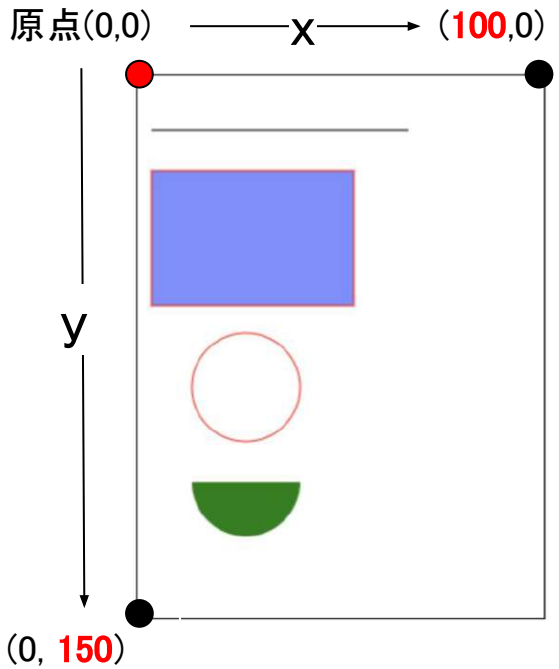
「2. 基本図形描画」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ 基本図形描画

2. 内容
   - ❑ 線・矩形・円の描画、図形の塗りつぶし

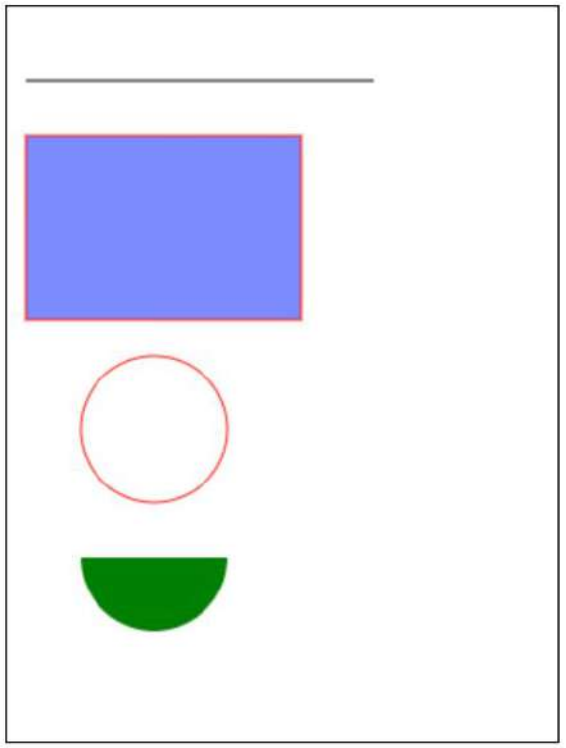
3. ポイント
   - ❑ 座標は、キャンバスの左上を原点 (0, 0) とし、x軸は左右、y軸は上下で指定する。
   - ❑ context.beginPath() 新しいパスを開始する
   - ❑ context.moveTo(x, y) 始点の設定 context.lineTo(x, y) 終点の指定 context.stroke() 線描画
   - ❑ context.fillRect(x, y, width, height) 四角形の塗りつぶし描画
   - ❑ context.strokeRect(x, y, width, height) 四角形の輪郭線描画
   - ❑ context.arc(x, y, radius, startAngle, endAngle [, counterclockwise]) 円弧の輪郭線描画
   - ❑ context.fill() 内部塗りつぶし



コードの例と解説「2. 基本図形描画」

https://t-cool.github.io/html5-level2-canvas/2basic_graphics.html

コードの例と解説「2. 基本図形描画」



```
<!DOCTYPE html>
<html>
<body>
<!-- 1. 描画用のキャンバス要素を作成する -->
<canvas id="myCanvas" width="300" height="400" style="border:1px s
<script>
// 2. 2Dコンテキストを初期化する
// canvas要素を取得する
```

コードの解説は解説動画をご覧ください。

```
context.fillStyle = 'rgba(0, 0, 255, 0.5)'; // 塗りつぶしの色
context.fillRect(10, 70, 150, 100); // 塗りつぶしの矩形
context.strokeStyle = 'red'; // 線の色
context.strokeRect(10, 70, 150, 100); // 線で囲まれた矩形

// 円を描画する
context.beginPath();
context.arc(80, 230, 40, 0, Math.PI * 2); // 円
context.stroke();

// 半円(円弧)を緑色で塗りつぶす
context.beginPath();
```

2.3.1 Canvas(2D) 解説

「3. テキスト描画」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ テキスト描写

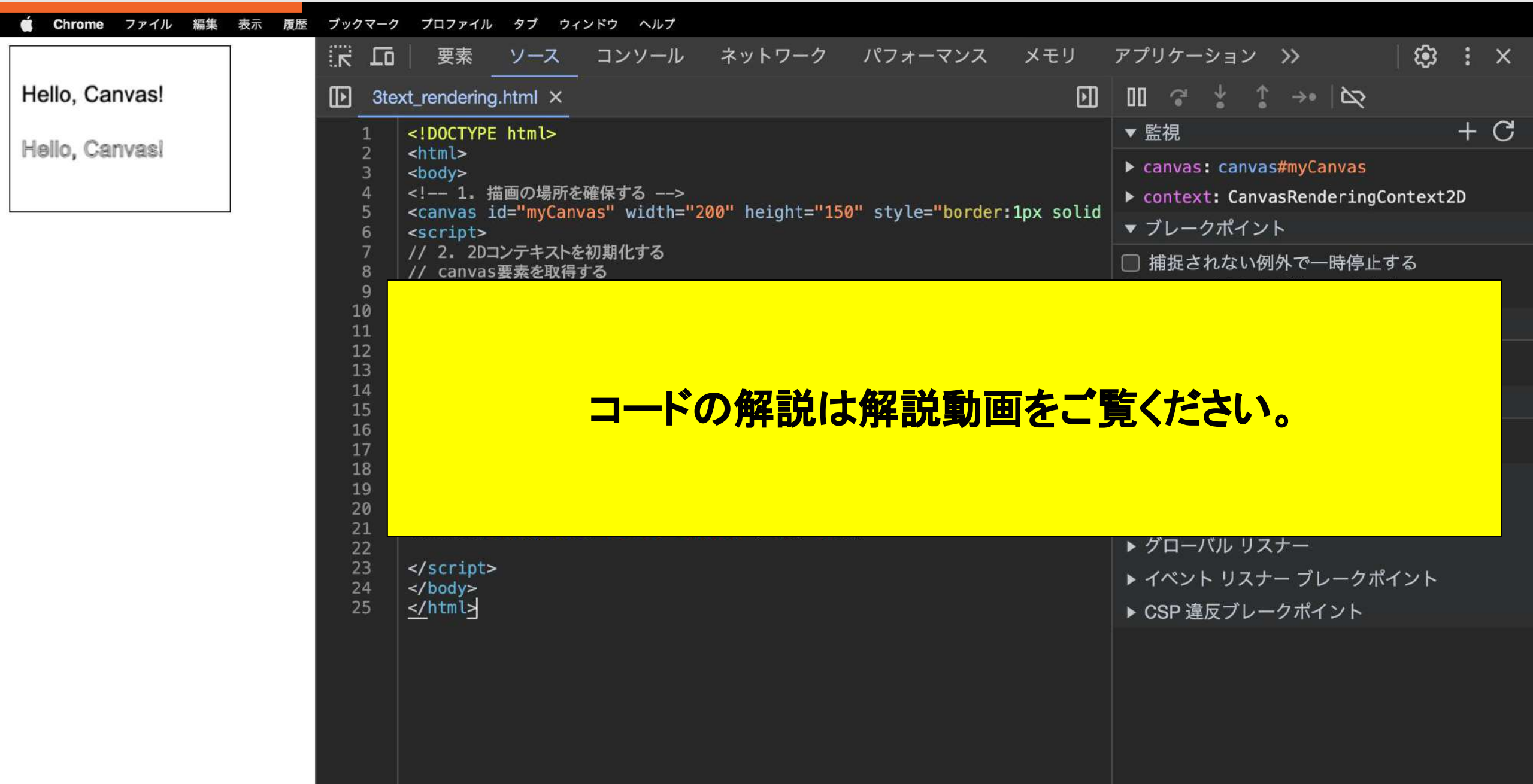
2. 内容
   - ❑ フォントの設定
   - ❑ テキストの塗りつぶし描画と輪郭描画

3. ポイント
   - ❑ context.font　フォントの種類や大きさを変更する
   - ❑ context.fillText(text, x, y)　テキストの塗りつぶし描画
   - ❑ context.strokeText(text, x, y)　テキストの輪郭描画

コードの例と解説「3. テキスト描画」

https://t-cool.github.io/html5-level2-canvas/3text_rendering.html

コードの例と解説「3. テキスト描画」



2.3.1 Canvas(2D) 解説

「4. 変形」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ 変形

2. 内容
   - ❑ 回転、変形の状態を元に戻す

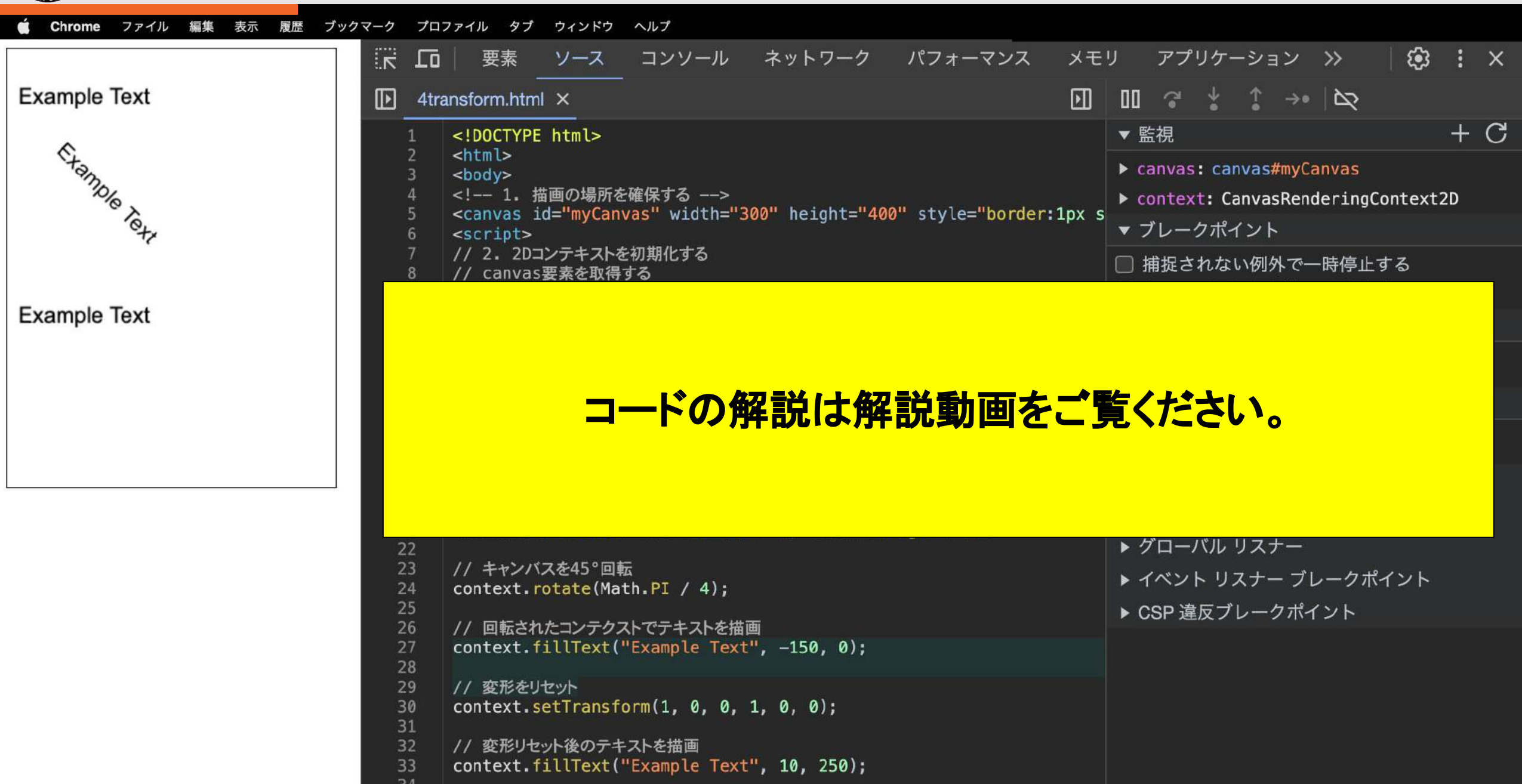
3. ポイント
   - ❑ context.translate(x,y)　原点を移動
   - ❑ context.rotate(angle)　原点を中心に回転する

     角度はラジアン角（Math.PI が 180°）で指定する。例：Math.PI/4 = 180/4 = 45
   - ❑ context.setTransform(1,0,0,1,0,0)　変形状態をリセットする

コードの例と解説「4. 変形」

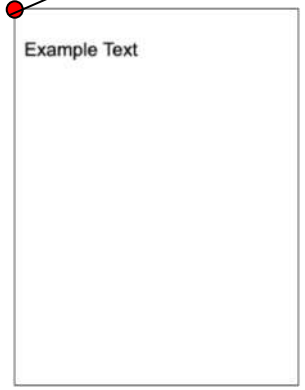
https://t-cool.github.io/html5-level2-canvas/4transform.html

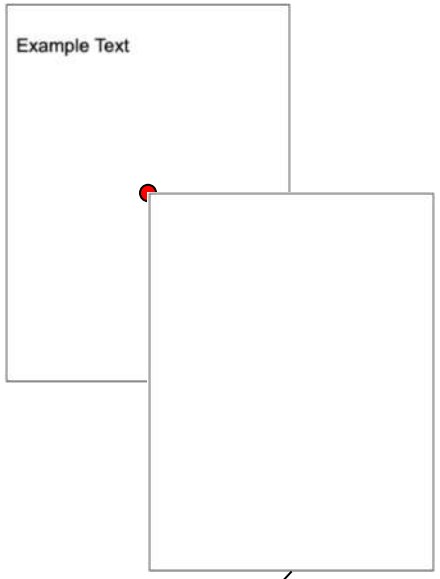
コードの例と解説「4. 変形」



【図解】回転後、変形の状態を戻し、文字を描写するまでの流れ

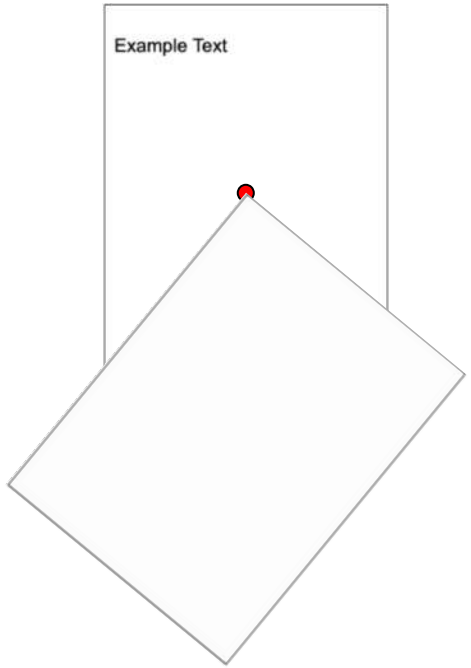
原点(0,0)





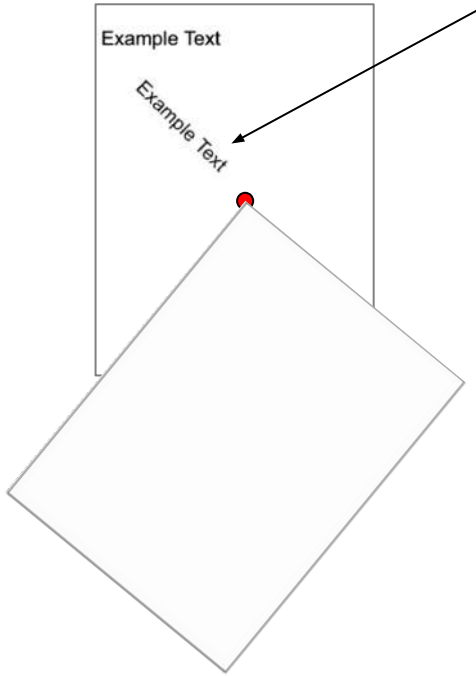
context.translate(canvas.width / 2,
canvas.height / 2);

分かりやすくするために枠
をつけています

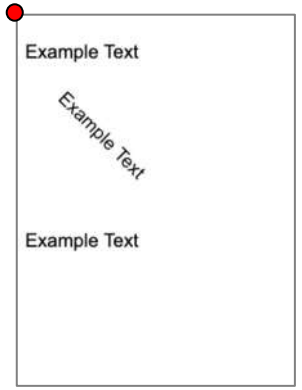


context.rotate(Math.PI / 4);

テキストの長さ分 x 座標を左に描写
するため -150 とする



context.fillText("Example Text", -150, 0);



context.setTransform(1, 0, 0, 1, 0, 0);

context.fillText("Example Text", 10, 250);

2.3.1 Canvas(2D) 解説

「5. エフェクト」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ エフェクト

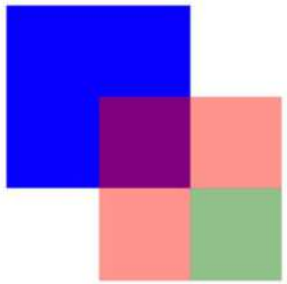
2. 内容
   - ❑ Canvasへの透明度指定、図形の合成

3. ポイント
   - ❑ context.globalAlpha　全描画の透明度設定
   - ❑ context.globalCompositeOperation　図形の合成方法の設定

コードの例と解説「5. エフェクト」

https://t-cool.github.io/html5-level2-canvas/4transform.html

コードの例と解説「5. エフェクト」



```
<!DOCTYPE html>
<html>
<body>
<!-- 1. 描画の場所を確保する -->
<canvas id="myCanvas" width="250" height="300" style="border:1px solid
<script>
// 2. 2Dコンテキストを初期化する
// canvas 要素を取得する
```

コードの解説は解説動画をご覧ください。

```
context.fillStyle = 'red';
context.fillRect(100, 100, 100, 100);

// 透明度をリセットする
context.globalAlpha = 1.0;

// 合成方法を設定する
context.globalCompositeOperation = 'source-atop';

// 図形を塗りつぶしで描画する(合成方法を適用)
context.fillStyle = 'green';
context.fillRect(150, 150, 100, 100);
```

2.3.1 Canvas(2D) 解説

「6. イメージデータ」

2.3.1 Canvas(2D) 解説

1. 項目名
   - ❑ イメージデータ

2. 内容
   - ❑ 画像の描画

3. ポイント
   - ❑ context.drawImage(image, dx, dy)　画像をキャンバスに描画する
     - → dx : destination x（目的地点の x座標）
     - → dy : destination y（目的地点の y座標）

コードの例と解説「 6. イメージデータ 」

https://t-cool.github.io/html5-level2-canvas/6imageData.html

コードの例と解説「 6. イメージデータ 」


```
<!DOCTYPE html>
<html>
<body>
<!-- 1. 描画の場所を確保する -->
<canvas id="myCanvas" width="300" height="300" style="border:1px s
<script>
// 2. 2Dコンテキストを初期化する
// canvas 要素を取得する
```

コードの解説は解説動画をご覧ください。

```
// オリジナル画像を描画する
context.drawImage(img, 50, 100);

// 座標点を描画する
context.beginPath();
context.arc(50, 100, 5, 0, Math.PI * 2);
context.fillStyle = 'red';
context.fill();

// 座標テキストを描画する
context.fillStyle = 'black';
```

さいごに

さいごに

HTML5 プロフェッショナル認定試験について

- レベル2では JavaScript とブラウザ API が中心。
- 試験範囲を参照し、体系的な学習が必要。

Canvas(2D) について

- コードの動作を視覚的に確認しやすいので、学習のスタート地点として最適。
- 試験範囲は2Dコンテキストのみ（WebGL は含まず）。
- Canvas (2D)の学習内容：コンテキスト、基本図形描画、テキスト描画、変形、エフェクト、イメージデータ

学習環境（開発環境）について

- 自身が使いやすいエディタの環境の構築が必須。VSCode 等、好みに合わせて選ぶ。
- ブラウザのデベロッパーツールの活用。「ソース」タブで JavaScriptの動作を確認できる。「ステップ実行」では、コードを1行ずつ実行し、動作の確認ができる。

参考資料

参考資料

書籍：

- 株式会社富士通ラーニングメディア, 抜山 雄一, 七條 怜子, 結城 陽平（2018）『HTML教科書 HTML5プロフェッショナル認定試験レベル2 スピードマスター問題集 Ver2.0対応』翔泳社

Web サイト：

- LPI Japan, “HTML5プロフェッショナル認定試験’, <https://html5exam.jp>
- WHATWG, “HTML Living Standard”, <https://html.spec.whatwg.org>
- Mozilla, “MDN web docs, JavaScript”, <https://developer.mozilla.org/ja/docs/Web/JavaScript>
- Mozilla, “MDN web docs, Canvas API”, <https://developer.mozilla.org/ja/docs/Web/API/Canvas_API>
- かわいいフリー素材集 いらすとや, <https://www.irasutoya.com>

JavaScript の学習におすすめの書籍：

- ES(ECMAScript) 2015以降
  - azu, Suguru Inatomi 著「JavaScript Primer 改訂2版 迷わないための入門書」KADOKAWA
  - あんどうやすし（2020）『ハンズオン JavaScript』 オライリー・ジャパン
- ECMAScript 2015以前：（プロトタイプベース言語としての特徴の把握）
  - 磯 博（2017）『徹底マスター JavaScript の教科書』SB Creative
  - Cody Lindley 著, 和田祐一郎 訳（2013）『開眼！JavaScript』オライリー・ジャパン
  - Nicholas C.Zakas 著, 和田祐一郎 訳（2014）『オブジェクト指向 JavaScript の原則』オライリー・ジャパン
  - Stoyan Stefanov 著, 豊福剛 訳（2011）『JavaScript パターン』オライリー・ジャパン