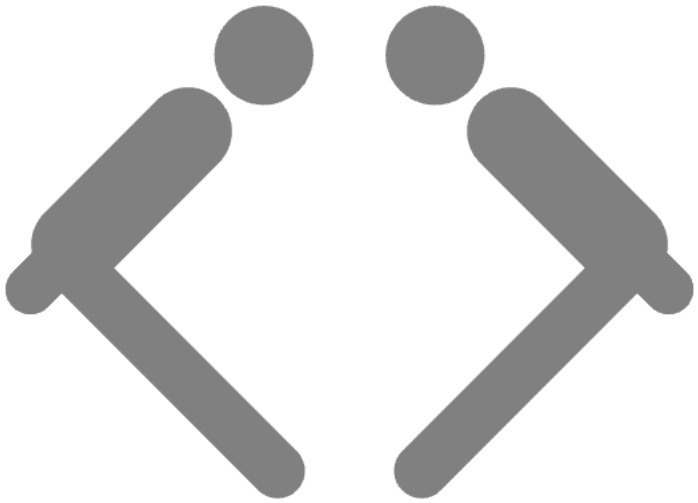

PICTAGRAMMING

(ピクタグラミング)

(2025 年 4 月 22 日版)

目次

0. ピクタグラミングについて ･･･ p3

1. ピクトグラムの表示 ･･･ p5

2. ピクトグラムエリアを変化させる ･･･ p7

3. 人型ピクトグラムを変化させる ･･･ p9

4. ピクトグラムに線，円，文字を追加 ･･･ p12

5. HPML への CSS の適用 ･･･ p15

6. HPML への CSS の適用（id） ･･･ p17

7. HPML への CSS の適用（class） ･･･ p19

8. HPML と HTML を組み合わせる ･･･ p21

9. 練習問題 ･･･ p23

10. オリジナルの作品制作 ･･･ p24

0. ピクタグラミングについて

概要

Web ページを作成するためには，プログラミング言語の他に **HTML (HyperText Markup Language)** というマークアップ言語と，**CSS (Cascading Style Sheets)** というスタイルシート言語が必要になります．HTML は Web ページの構造を作成するため，CSS は Web ページの見た目を整えるために使用します．ピクタグラミングは，ピクトグラムの作成を通じて HTML と CSS の記法について学習することができるアプリケーションです．

ピクタグラミングでは，**HPML (Human Pictogram Markup Language)** というピクトグラムを作成するための独自のマークアップ言語を定義しており，HPML を記述することで様々なピクトグラムを作成することができます．

ピクトグラムとは

ピクトグラムとは，「非常口」に代表されるグラフィックシンボルであり，図 0-1 が一例です．ピクトグラムを使うことで，情報を視覚的に伝えることが可能です．「非常口」のピクトグラムに描かれている“人の形状をしたピクトグラム”は「人型ピクトグラム」と呼ばれます．

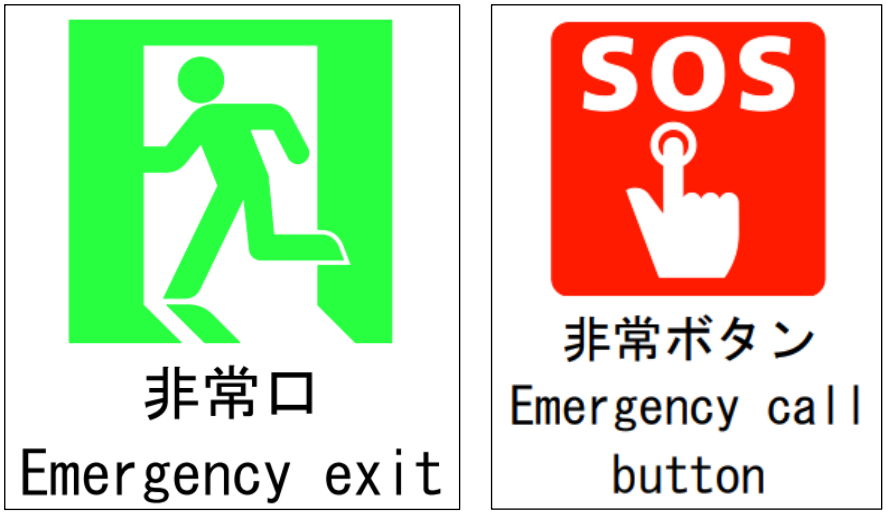

図 0-1. ピクトグラムの例（国土交通省ホームページより引用）

ピクトグラムは学校や商業施設，道路など，様々な場所で見ることができます．年齢や国籍などに関わらず，情報を伝えることができる手段として注目されています．

アクセス

ピクタグラミングは以下の URL からアクセスすることができます．
https://pictogramming.org/apps/pictagramming/

画面説明

図 0-2 がピクタグラミングの画面のスクリーンショットです．

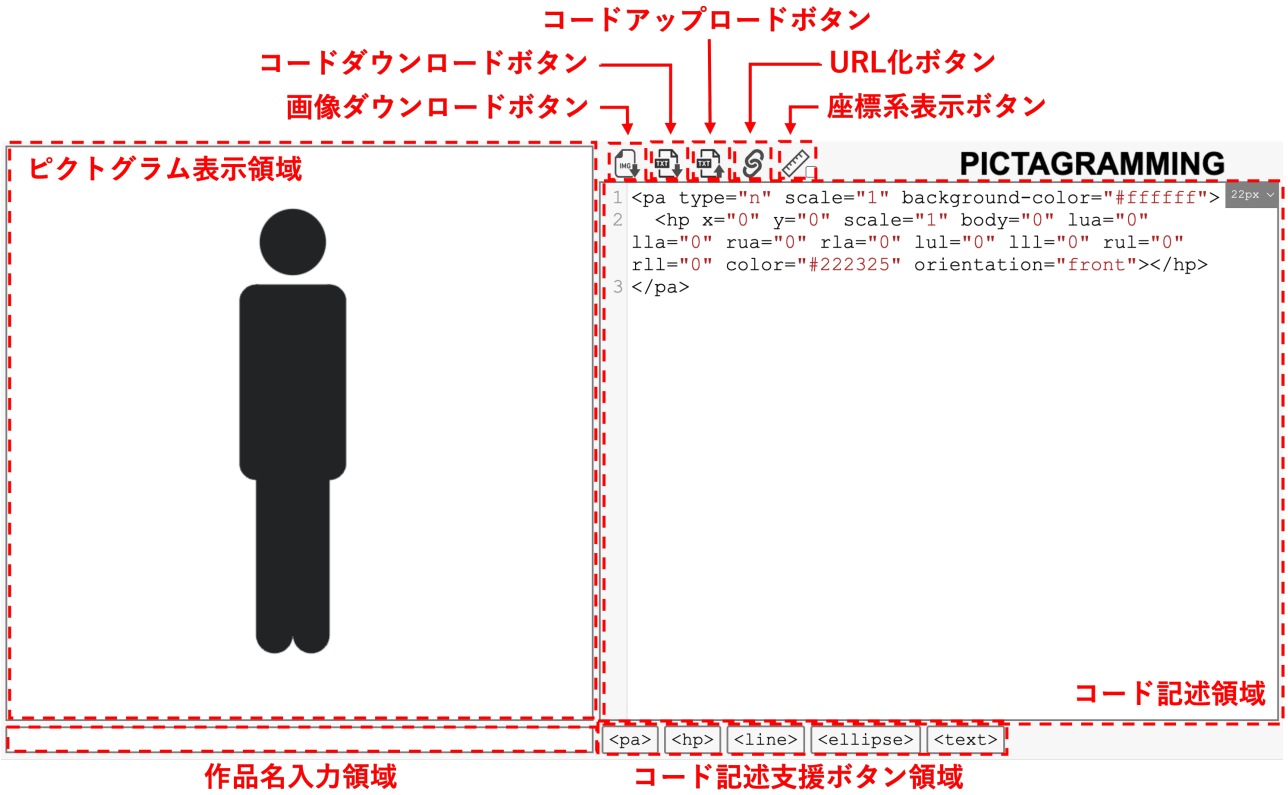

図 0-2. ピクタグラミングのスクリーンショット

画面左上部はコードの実行結果を表示するピクトグラム表示領域，右中央部はコードを入力するコード記述領域，左下部は作品名を入力する作品名入力領域，右下部はコードの入力を支援するコード記述支援ボタン領域です．また，画面右上部にある 5 つのボタンの機能について表 0-1 に示します．

表 0-1. ピクタグラミングの画面説明

画像	名称	説明
	画像ダウンロードボタン	ピクトグラム表示領域を画像としてダウンロードする．
	コードダウンロードボタン	コード記述領域のコードをダウンロードする．
	コードアップロードボタン	コード記述領域にコードをアップロードする．
	URL 化ボタン	コード記述領域と作品名入力領域の情報を含んだ URL をクリップボードにコピーする．
	座標系表示ボタン	ピクトグラム表示領域に座標系を表示する．

さっそくピクタグラミングを始めてみましょう．

1. ピクトグラムの表示

タグとは

HTML や HPML では，タグを組み合わせることで，様々なピクトグラムコンテンツを作成することができます．タグは以下のような形で記述します．

<タグ名 属性名="属性値">	タグのコンテンツ	</タグ名>
開始タグ	内容	終了タグ

タグは開始タグ，コンテンツ，終了タグから成り立っており，コンテンツの中にはさらにタグを書くこともできます．また，開始タグの中には，属性というものを記述することができます．ピクタグラミングでタグについて学んでみましょう．

ピクタグラミングにアクセスすると，図 1-1 のような画面が表示されます．

図 1-1. ピクタグラミング画面（初期状態）

コード記述領域には，「コード例 1」のコードが入力されています．

コード例 1　（←クリックするとピクタグラミングでコード例を開きます）

```
<pa type="n" scale="1" background-color="#ffffff">
  <hp x="0" y="0" scale="1" body="0" lua="0" lla="0" rua="0" rla="0" lul="0" lll="0" rul="0" rll="0" color="#222325" orientation="front"></hp>
</pa>
```

コード例 1 は pa タグと hp タグで構成されています．「type="n"」といった属性の記述が複数されていますが，いったん無視してください．pa タグは Pictogram Area（ピクトグラムエリア）を，hp タグは Human Pictogram（人型ピクトグラム）を意味するタグです．

ピクトグラムを表示するためには，まず pa タグを記述する必要があります．pa タグを記述することで，ピクトグラムを表示させるエリアを指定できます．そして，pa タグのコンテンツに hp タグを記

述することで，人型ピクトグラムを表示させることができます．表 1-1 も参考にしてください．

表 1-1．HPML タグセット

タグセット	処理
<pa>	Pictogram Area（ピクトグラムエリア）を設置する．
<hp>	pa タグの中に記述することで，Human Pictogram（人型ピクトグラム）を表示する．

なお，このように pa タグの中に hp タグを記述したとき，pa タグを親要素，hp タグを子要素といいます．hp タグだけで人型ピクトグラムを表示させることはできず，pa タグが親要素である必要があります．

使用するタグや属性を変更，追加することで，ピクトグラムを変化させることができます．次の章で見ていきましょう．

2. ピクトグラムエリアを変化させる

1 章では，pa タグと hp タグについて見てきました．2 章では，ピクトグラムのタイプを変更したいと思います．タイプを変更するには pa タグの type 属性を使用します．初期状態では「type="n"」と type 属性には"n"が指定されていますが，これはピクトグラムエリアのタイプを Normal（標準）にするという意味です．

"n"を，"p"や"a"，"i"，"s"，"sg"，"sr"，"rv"にすることもできます．これらは，ピクトグラムエリアをそれぞれ，Prohibit（禁止），Attention（注意），Instruction（指示），Safety（安全），Safety Green（安全（緑）），Safety Red（安全（赤）），Reverse（反転）にするという意味です．type 属性の値を"n"から"a"に変えてみましょう（「コード例 2」）．

コード例 2

```
<pa type="a" scale="1" background-color="#ffffff">
  <hp x="0" y="0" scale="1" body="0" lua="0" lla="0" rua="0" rla="0" lul="0"
lll="0" rul="0" rll="0" color="#222325" orientation="front"></hp>
</pa>
```

"n"を"a"に変えると，図 2-1 のように Attention（注意）のマークが表示されます．

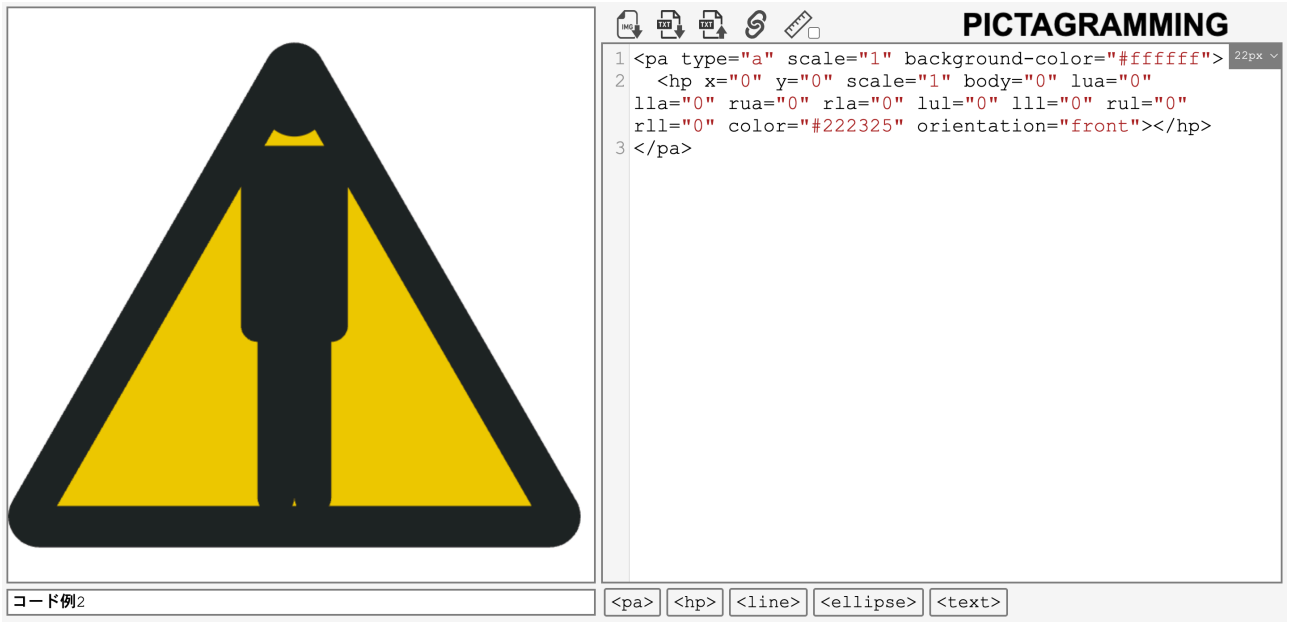

図 2-1. ピクタグラミングの画面（コード例 2）

pa タグには，type 属性の他に，scale 属性と background-color 属性もあります．scale 属性はピクトグラムエリアの大きさを変更する属性，background-color 属性はピクトグラムエリアの背景色を変更する属性です．自由に値を変化させてみましょう．

表 2-1 に pa タグの属性一覧を示します．

表 2-1. pa タグの属性一覧

属性	とりうる値	処理	初期値	備考
type="arg1"	"n", "p", "a", "i", "s", "sg", "sr", "rv"	タイプを arg1 が"n"のとき Normal（標準），"p"のとき Prohibit（禁止），"a"のとき Attention（注意），"i"のとき Instruction（指示），"s"のとき Safety（安全），"sg"のとき Safety Green（安全緑），"sr"のとき Safety Red（安全赤），"rv"のとき Reverse（反転）にする.	"n"	図 2-2 を参照
scale="arg1"	正の数	幅，高さともに arg1×640 ピクセルにする.	"1"	-
background-color="arg1"	色キーワード，16 進数文字列，RGB 関数など	背景色を arg1 にする. type 属性の値が"n", "p", "a", "i", "s"のいずれかの場合のみ有効であり，"sg", "sr", "rv"の場合無効である.	"#ffffff"	下記「色の指定方法について」を参照

ピクトグラムエリアのタイプについて

pa タグの type 属性の値を変化させると，ピクトグラムエリアのタイプを変更させることができます. 図 2-2 に，ピクトグラムエリアのタイプ一覧を示します.

図 2-2. ピクトグラムエリアのタイプ一覧

色の指定方法について

HTML や CSS で色を指定する方法には，色キーワードや 16 進数文字列，RGB 関数などがあります. これはピクタグラミングにおいても同様です.

例えば，赤色を指定したい場合，色キーワードなら"red"と指定します. ですが，"red"は 16 進数文字列や RGB 関数でも表現できます. 16 進数文字列なら"#ff0000"，RGB 関数なら"rgb(255, 0, 0)"とすれば，色キーワードで"red"で指定したことと同じになります.

詳しくは，教科書や MDN のドキュメント（https://developer.mozilla.org/ja/docs/Web/CSS/CSS_colors/Applying_color）等を参照してください.

なお，検索エンジンで「カラー選択ツール」というように検索すると，色を簡単に選択することができるツールが出てきますので，必要に応じて試してみてください.

3. 人型ピクトグラムを変化させる

ここまでは，pa タグの属性を変化させることでピクトグラムエリアを変化させてきました．3 章では，hp タグの属性を変化させることで人型ピクトグラムを変化させたいと思います．

hp タグには，位置や大きさ，身体や手足の角度，色，向きを変更する属性があります．hp タグの属性の値を変化させてみましょう（「コード例 3」）．

コード例 3

```
<pa type="a" scale="1" background-color="#ffffff">
  <hp x="20" y="70" scale="0.6" body="60" lua="-120" lla="45" rua="45"
rla="90" lul="20" lll="-60" rul="45" rll="-60" color="#222325"
orientation="front"></hp>
</pa>
```

すると，図 3-1 のようにピクトグラムが変化します．人型ピクトグラムが転んでいるかのようになりました．

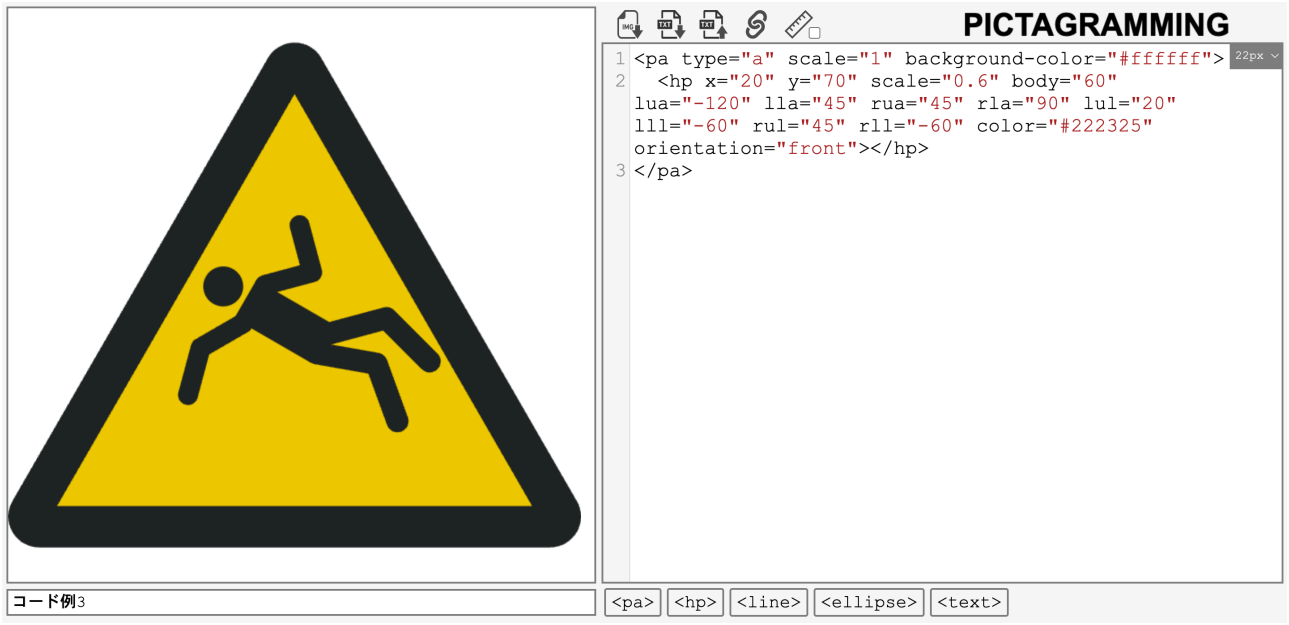

図 3-1. ピクタグラミングの画面（コード例 3）

x 属性と y 属性はそれぞれ水平方向，垂直方向の位置を指定する属性です．また，scale 属性は大きさを指定する属性です．そして，body 属性，lua 属性，lla 属性，rua 属性，rla 属性，lul 属性，lll 属性，rul 属性，rll 属性は，それぞれピクトグラムの体，左肩，左肘，右肩，右肘，左股，左膝，右股，右膝の角度を指定する属性です．

表 3-1 に hp タグの属性一覧を示します．

表 3-1. hp タグの属性一覧

属性	とりうる値	処理	初期値	備考
x="arg1", y="arg2"	数値	座標（arg1, arg2）を中心座標にする.	ともに"0"	図 3-2 を参照
scale="arg1"	正の数	大きさを標準（初期状態）の arg1 倍にする.	"1"	-
body="arg1", lua="arg2", lla="arg3", rua="arg4", rla="arg5", lul="arg6", lll="arg7", rul="arg8", rll="arg9"	数値	体を arg1 度，左肩（Left Upper Arm）を arg2 度，左肘（Left Lower Arm）を arg3 度，右肩（Right Upper Arm）を arg4 度，右肘（Right Lower Arm）を arg5 度，左股（Left Upper Leg）を arg6 度，左膝（Left Lower Leg）を arg7 度，右股（Right Upper Leg）を arg8 度，右膝（Right Lower Leg）を arg9 度反時計回りに回転させる.	全て"0"	図 3-3 を参照
orientation="arg1"	"front"，または"side"	arg1 が"front"の場合は正面向きに，"side"の場合は横向きにする.	"front"	-
color="arg1"	色キーワード，16 進数文字列，RGB 関数など	色を arg1 にする.	pa タグの type 属性の値が"n"，"p"，"a"のときは"#222325"，"i"，"sg"，"sr"，"rv"のときは"#eeedef"，"s"のときは"#009c64"	2 章「色の指定方法について」を参照

ピクトグラムエリアの座標について

表 3-1 にあるように，hp タグに x 属性や y 属性を指定することで人型ピクトグラムの位置を指定することができます．ここで，ピクトグラムエリアの座標の考え方を示します.

ピクタグラミングでは，pa タグで表したピクトグラムエリアの上下左右の中心を x="0"，y="0"の地点とします．そして，ピクトグラムエリアの左右の端をそれぞれ x="-320"，x="320"の地点とし，上下の端をそれぞれ y="-320"，y="320"の地点とします．これは pa タグの scale 属性の値が変化しても変化しません．つまり，ピクトグラムエリアの大きさに関わらず，座標は同じです（例えば，pa タグの scale 属性の値が"1"でも"0.5"でも，ピクトグラムエリアの右上の点は x="320"，y="-320"の地点となります）．図 3-2 に座標の考え方を図にしたものを示します.

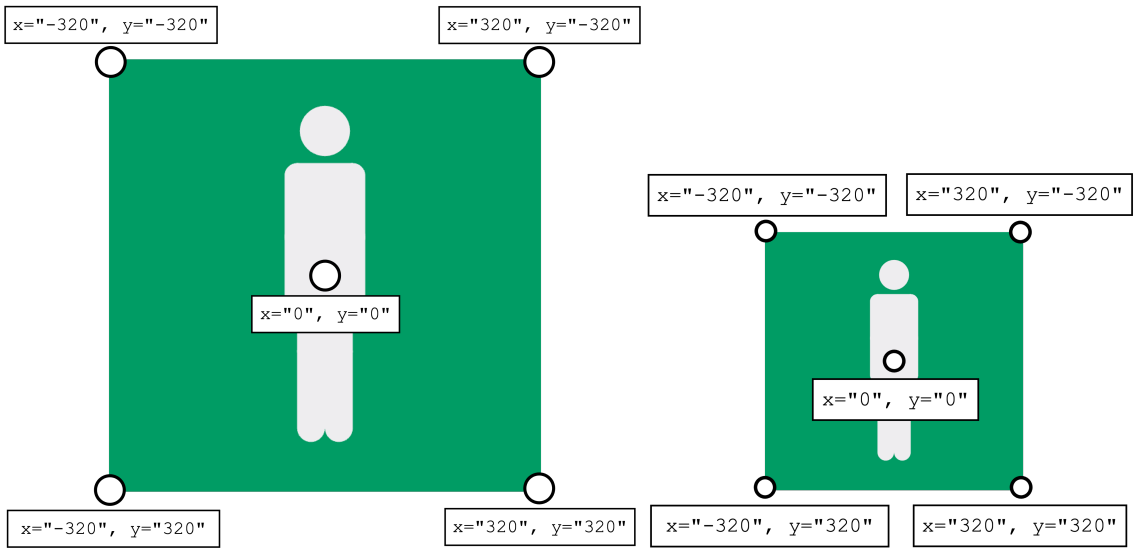

図 3-2. ピクトグラムエリアの座標について
（ピクトグラムエリアの大きさが変化しても，座標は同じ）

この座標の考え方は，hp タグに関わらず，次のセクションから出てくる HPML タグに関しても同様です．困ったときには，座標系表示ボタンを使用すると，座標系がわかりやすいです．

部位について

図 3-3 に人型ピクトグラムのパーツの一覧を示します．体は body です．手足は英字 3 字で表現します．1 文字目は左（Left）か右（Right）か，2 文字目は上側（Upper）か下側（Lower）か，3 文字目は腕（Arm）か足（Leg）かです．それぞれ対応する属性で，角度を変化させることができます．

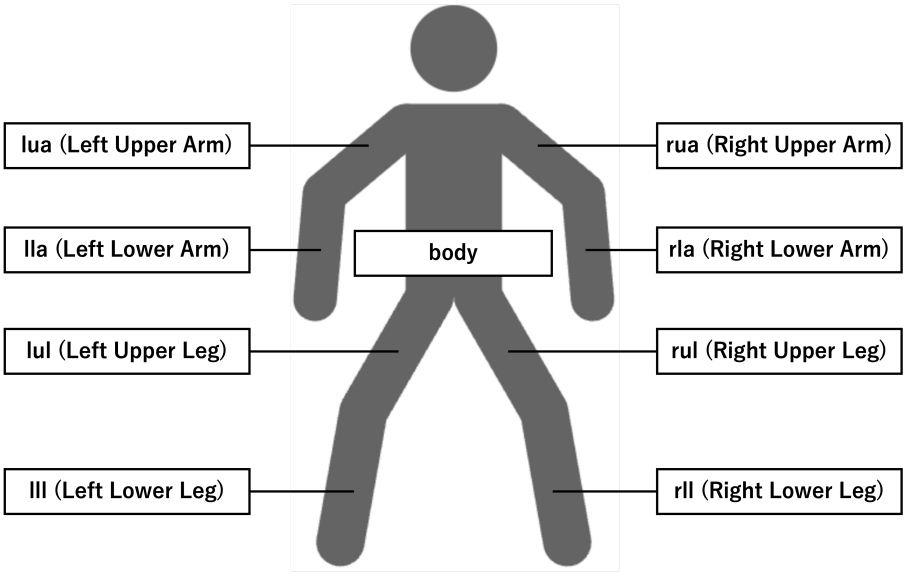

図 3-3. 人型ピクトグラムの身体の部位の名称

4. ピクトグラムに線，楕円，円，文字を追加

HPML タグには，線や楕円，円，テキストを描画するための line タグや ellipse タグ，circle タグ，text タグもあります．4 章では，line タグを用いて段差を作成してみましょう（「コード例 4」）．

コード例 4（example-4.txt）

```
<pa type="a" scale="1" background-color="#ffffff">
  <hp x="20" y="70" scale="0.6" body="60" lua="-120" lla="45" rua="45" rla="90"
lul="20" lll="-60" rul="45" rll="-60" color="#222325" orientation="front"></hp>
  <line x1="-160" y1="150" x2="80" y2="150" width="20" color="#222325"></line>
  <line x1="80" y1="140" x2="80" y2="210" width="20" color="#222325"></line>
  <line x1="80" y1="200" x2="160" y2="200" width="20" color="#222325"></line>
</pa>
```

3 つの line タグを加えることで，図 4-1 のように段差を表現でき，段差で転倒しないよう注意を促すピクトグラムを作成できます．

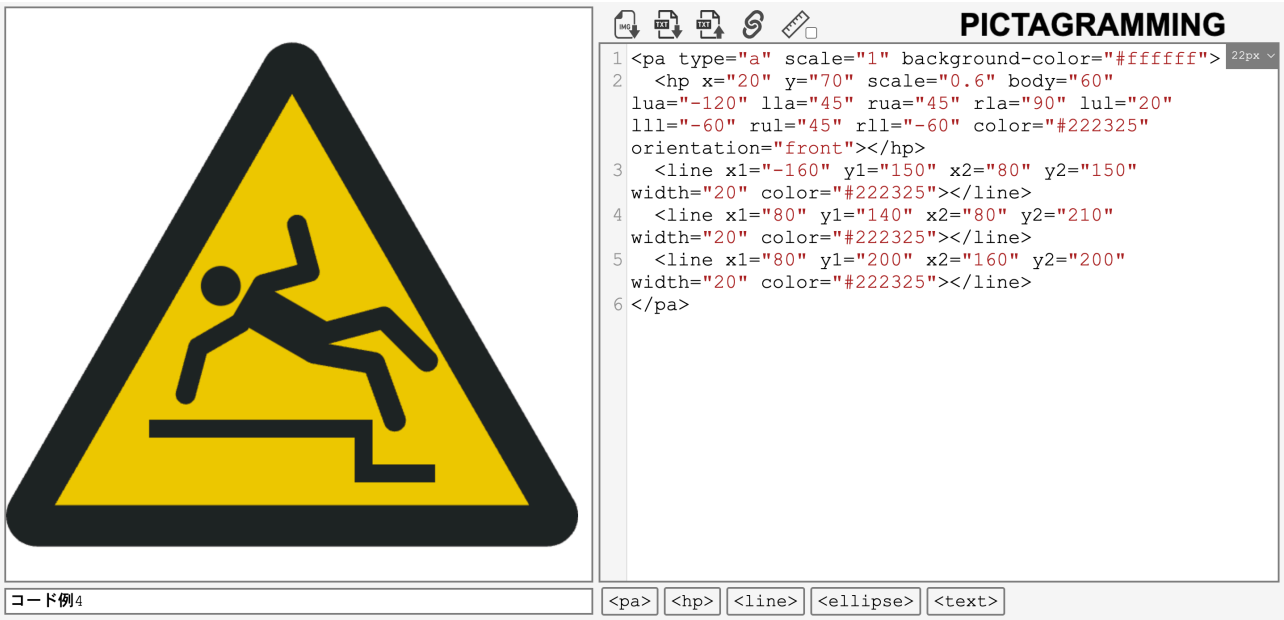

図 4-1. ピクタグラミングの画面（コード例 4）

コード例 4 では，pa タグの中に 3 つの line タグを加えましたが，line タグ以外のタグを pa タグの中に複数入れることも可能です．例えば，pa タグの中に hp タグを 2 つ加えると，人型ピクトグラムが 2 体表示されます．

表 4-1 に HPML のタグセットを，表 4-2 から 4-5 に line タグ，ellipse タグ，circle タグ，text タグの属性一覧を示します．

表 4-1. HPML タグセット

タグセット	処理
<line>	pa タグの中に記述することで，線を描画する．
<ellipse>	pa タグの中に記述することで，楕円を描画する．
<circle>	pa タグの中に記述することで，円を描画する．
<text>	pa タグの中に記述することで，テキストを描画する．

表 4-2. line タグの属性一覧

属性	とりうる値	処理	初期値	備考
x1="arg1", y1="arg2", x2="arg3", y2="arg4"	数値	座標（arg1, arg2）から，座標（arg3, arg4）を描画範囲にする.	全て"0"	図 3-2 を参照
width="arg1"	正の数	太さを arg1 にする.	"20"	-
color="arg1"	色キーワード，16 進数文字列，RGB 関数など	色を arg1 にする.	pa タグの type 属性の値が"n", "p", "a"のときは"#222325", "i", "sg", "sr", "rv"のときは"#eeedef", "s"のときは"#009c64"	2 章「色の指定方法について」を参照

表 4-3. ellipse タグの属性一覧

属性	とりうる値	処理	初期値	備考
x="arg1", y="arg2"	数値	座標（arg1, arg2）を中心座標にする.	ともに"0"	図 3-2 を参照
width="arg1", height="arg2"	正の数	幅を arg1，高さを arg2 にする.	ともに"50"	-
angle="arg1"	数値	arg1 度反時計回りに回転させる.	"0"	-
color="arg1"	色キーワード，16 進数文字列，RGB 関数など	色を arg1 にする.	pa タグの type 属性の値が"n", "p", "a"のときは"#222325", "i", "sg", "sr", "rv"のときは"#eeedef", "s"のときは"#009c64"	2 章「色の指定方法について」を参照

表 4-4. circle タグの属性一覧

属性	とりうる値	処理	初期値	備考
x="arg1", y="arg2"	数値	座標（arg1, arg2）を中心座標にする.	ともに"0"	図 3-2 を参照
r="arg1"	正の数	半径を arg1 にする.	"50"	-
color="arg1"	色キーワード，16 進数文字列，RGB 関数など	色を arg1 にする.	pa タグの type 属性の値が"n", "p", "a"のときは"#222325", "i", "sg", "sr", "rv"のときは"#eeedef", "s"のときは"#009c64"	2 章「色の指定方法について」を参照

表 4-5. text タグの属性一覧

属性	とりうる値	処理	初期値	備考
x="arg1", y="arg2"	数値	座標（arg1, arg2）を描画の左下の点にする.	ともに"0"	図 3-2 を参照
font-size="arg1"	正の数	フォントの大きさを arg1 にする.	"50"	2 章「色の指定方法について」を参照
font-family="arg1"	フォントファミリー	フォントファミリーを arg1 にする.	"Hiragino Kaku Gothic ProN"	-
color="arg1"	色キーワード, 16 進数文字列, RGB 関数など	色を arg1 にする.	pa タグの type 属性の値が"n", "p", "a"のときは"#222325", "i", "sg", "sr", "rv"のときは"#eeedef", "s"のときは"#009c64"	下記「font-family 属性について」を参照

font-family 属性について

font-family 属性には, "Arial"などのフォントの名称を指定することができます. 詳細は MDN のドキュメント（https://developer.mozilla.org/ja/docs/Web/CSS/font-family）等を参照してください.

5. HPML への CSS の適用

今までは HPML の属性を用いてピクトグラムを作成してきました．ですが，ピクタグラミングでは HPML に CSS を適用してピクトグラムを作成することもできます．

CSS を適用するには，以下のような記述をする必要があります．

```
セレクター {
  プロパティ: プロパティ値;
}
```

セレクターとは，どの HPML タグに対して，CSS を適用するか指定するものです．例えば，hp タグに対して CSS を適用したい場合，セレクターには hp が入ります．

プロパティとはスタイルの種類のことです．CSS には多くのプロパティがありますが，ピクタグラミングでは CSS の学習を短時間で行えるようにするため，HPML にのみ使用できるプロパティを用意しています．プロパティとプロパティ値は，HPML タグの属性名と属性値と同様で，処理も同様です．

なお，HPML の属性を用いても CSS を適用しても，できる処理は同じですが，両方記述した場合，HPML の属性による指定が優先されます．なお，CSS は style タグの中に記述することで適用することができます．コード例 4 の pa タグと hp タグを CSS で書き換えてみます（「コード例 5」）．

コード例 5

```
<pa type="a" scale="1" background-color="#ffffff">
  <hp x="20" y="70" scale="0.6" body="60" lua="-120" lla="45" rua="45" rla="90"
lul="20" lll="-60" rul="45" rll="-60" color="#222325" orientation="front"></hp>
  <line x1="-160" y1="150" x2="80" y2="150" width="20" color="#222325"></line>
  <line x1="80" y1="140" x2="80" y2="210" width="20" color="#222325"></line>
  <line x1="80" y1="200" x2="160" y2="200" width="20" color="#222325"></line>
</pa>

<style>
pa {
  type: a;
  scale: 1;
  background-color: #ffffff;
}
hp {
  x: 20;
  y: 70;
  scale: 0.6;
  body: 60;
  lua: -120;
  lla: 45;
  rua: 45;
  rla: 90;
  lul: 20;
  lll: -60;
  rul: 45;
  rll: -60;
  color: #222325;
  orientation: front;
}
</style>
```

図 5-1 のように，HPML の属性を CSS に書き換えても，図 4-1 と同じピクトグラムになります.

図 5-1. ピクタグラミングの画面（コード例 5）

hp タグの属性ではなく，CSS で書き換えることができました.

6. HPML への CSS の適用（id）

ここまでで，hp タグに対して CSS を適用することができました．次は line タグに対して CSS を適用しましょう．ですが，今までのコードを見てわかるように，line タグは 3 つ記述しており，それぞれ指定したい値が異なります．そこで，**id** 属性を使用します．

<タグ名 **id="id の値"**> タグのコンテンツ </タグ名>
開始タグ　　終了タグ

id 属性は，コード全体で一意である必要があります．つまり，複数のタグに対して，同じ id の値をつけることができないということです．

id 属性の値の前にハッシュ記号（#）をつけることで，CSS のセレクターとすることもできます． 3 つの line タグにそれぞれ異なる id 属性を指定し，line タグに対しても CSS を適用します（「コード例 6」）．

コード例 6

```
<pa>
  <hp></hp>
  <line  id="line-top"  x1="-160"  y1="150"  x2="80"  y2="150"  width="20"
color="#222325"></line>
  <line  id="line-center"  x1="80"  y1="140"  x2="80"  y2="210"  width="20"
color="#222325"></line>
  <line  id="line-bottom"  x1="80"  y1="200"  x2="160"  y2="200"  width="20"
color="#222325"></line>
</pa>

<style>
pa {
  type: a;
  scale: 1;
  background-color: #ffffff;
}
hp {
  x: 20;
  y: 70;
  scale: 0.6;
  body: 60;
  lua: -120;
  lla: 45;
  rua: 45;
  rla: 90;
  lul: 20;
  lll: -60;
  rul: 45;
  rll: -60;
  color: #222325;
  orientation: front;
}
#line-top {
  x1: -160;
  y1: 150;
  x2: 80;
  y2: 150;
  width: 20;
  color: #222325;
```

```
}
#line-center {
  x1: 80;
  y1: 140;
  x2: 80;
  y2: 210;
  width: 20;
  color: #222325;
}
#line-bottom {
  x1: 80;
  y1: 200;
  x2: 160;
  y2: 200;
  width: 20;
  color: #222325;
}
</style>
```

id 属性を使用したことで，図 6-1 のように 3 つの line タグに対して個別に CSS を適用できました．

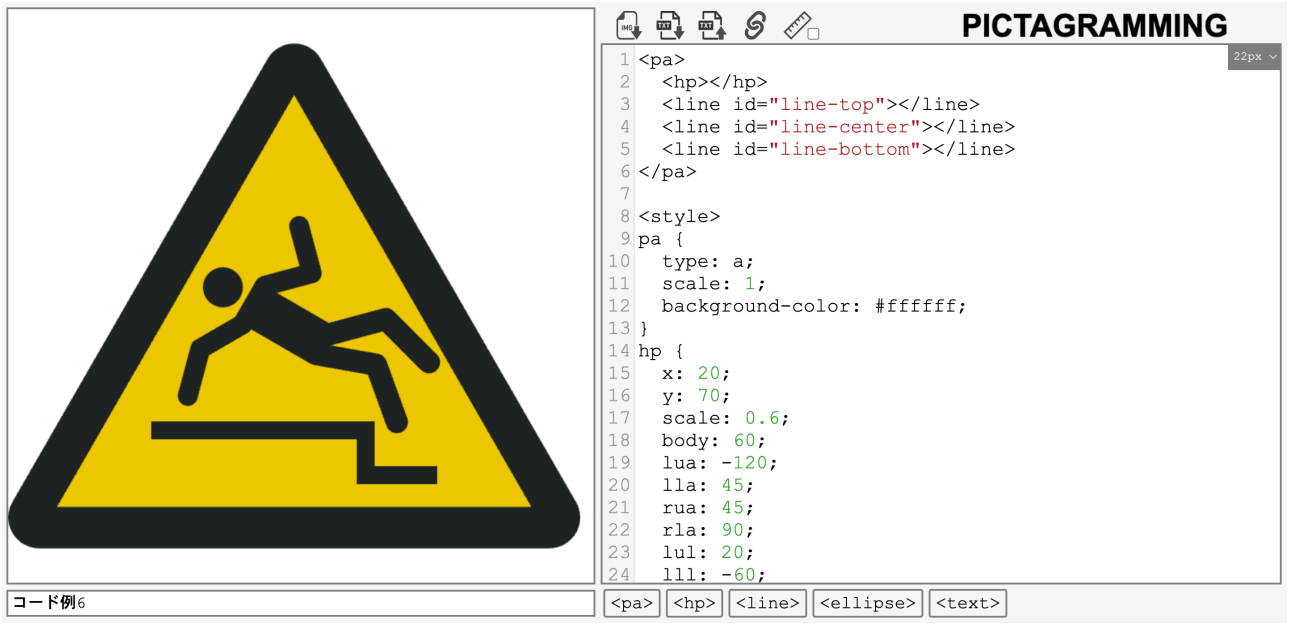

図 6-1. ピクタグラミングの画面（コード例 6）

次の章では，id 属性とは違う属性を利用した CSS の適用について見ていきます．

7. HPML への CSS の適用（class）

6 章では，3 つの line タグに対して．個別に CSS を適用することができました．7 章では，3 つの line タグ全てに対して CSS を適用することができる **class** 属性を使います．

<タグ名 **class="class の値"**> タグの内容 </タグ名>
開始タグ　　終了タグ

id 属性は，コード全体で一意である必要がありましたが，class 属性は一意である必要がありません．複数のタグに対して同じ class 属性の値をつけることで，一度に CSS を適用することが可能です．

class 属性の値の前にピリオド（.）をつけることで，CSS のセレクターとすることができます．人型ピクトグラムの色と線の色をどちらも灰色に変更したくなったと仮定して，class 属性を使用して CSS を適用してみましょう（「コード例 7」）．

コード例 7

```
<pa>
  <hp class="gray"></hp>
  <line id="line-top" class="gray"></line>
  <line id="line-center" class="gray"></line>
  <line id="line-bottom" class="gray"></line>
</pa>

<style>
pa {
  type: a;
  scale: 1;
  background-color: #ffffff;
}
hp {
  x: 20;
  y: 70;
  scale: 0.6;
  body: 60;
  lua: -120;
  lla: 45;
  rua: 45;
  rla: 90;
  lul: 20;
  lll: -60;
  rul: 45;
  rll: -60;
  color: #222325;
  orientation: front;
}
#line-top {
  x1: -160;
  y1: 150;
  x2: 80;
  y2: 150;
  width: 20;
  color: #222325;
}
#line-center {
  x1: 80;
  y1: 140;
  x2: 80;
```

```
  y2: 210;
  width: 20;
  color: #222325;
}
#line-bottom {
  x1: 80;
  y1: 200;
  x2: 160;
  y2: 200;
  width: 20;
  color: #222325;
}
.gray {
  color: gray;
}
</style>
```

すると，図 7-1 のように人型ピクトグラムと線の色を灰色にすることができます．

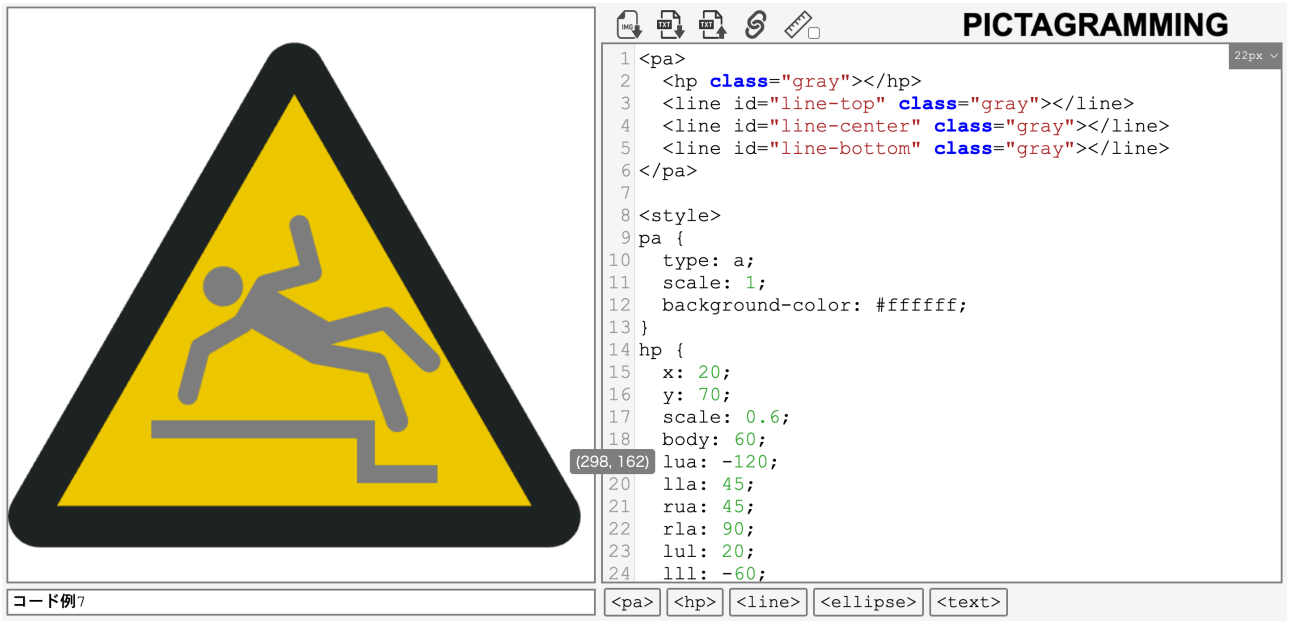

図 7-1. ピクタグラミングの画面（コード例 7）

複数の HPML タグに対して同じ CSS を適用したいとき，1 つずつ指定すると時間がかかってしまいます．ですが，class 属性を利用することで，同じような記述を何度もする必要がなくなります．このように，CSS を使用する利点は，再利用性の高いコードを書くことができるということです．

8. HPML と HTML を組み合わせる

ここまでは HPML を用いて段差に注意を促すピクトグラムを作成してきましたが，ピクタグラミングでは，HPML を HTML と組み合わせることも可能です.

コード例 6 に見出しを意味する h1 タグを組み合わせてみましょう（「コード例 8」）.

コード例 8

```
<h1>転倒に注意してください</h1>
<pa>
  <hp></hp>
  <line id="line-top"></line>
  <line id="line-center"></line>
  <line id="line-bottom"></line>
</pa>

<style>
h1 {
  text-align: center;
  font-size: 50px;
  margin: 0;
}
pa {
  type: at;
  scale: 1;
  background-color: #ffffff;
}
hp {
  x: 20;
  y: 70;
  scale: 0.6;
  body: 60;
  lua: -120;
  lla: 45;
  rua: 45;
  rla: 90;
  lul: 20;
  lll: -60;
  rul: 45;
  rll: -60;
  color: #222325;
  orientation: front;
}
#line-top {
  x1: -160;
  y1: 150;
  x2: 80;
  y2: 150;
  width: 20;
  color: #222325;
}
#line-center {
  x1: 80;
  y1: 140;
  x2: 80;
  y2: 210;
  width: 20;
  color: #222325;
```

```
}
#line-bottom {
  x1: 80;
  y1: 200;
  x2: 160;
  y2: 200;
  width: 20;
  color: #222325;
}
</style>
```

すると，図 8-1 のように，ピクトグラムに加えて，見出しのテキストが表示されます．

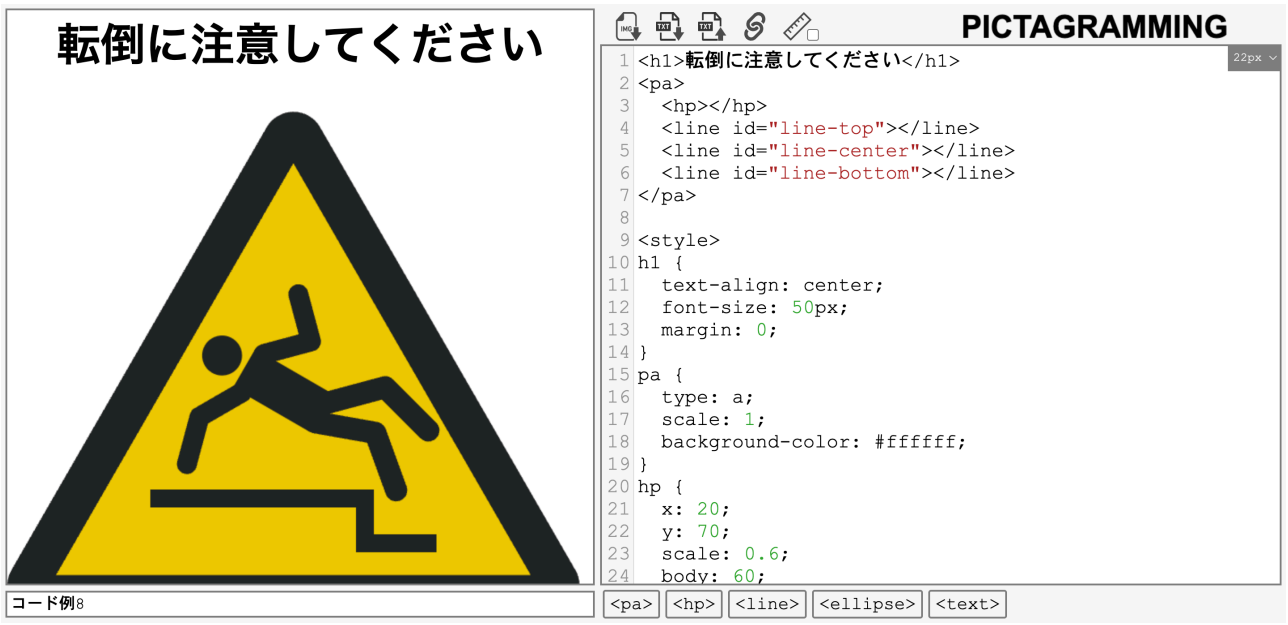

図 8-1. ピクタグラミングの画面（コード例 8）

HPML タグに，HTML タグを組み合わせることで，ピクトグラムを含んだコンテンツを手軽に作成することができます．

9. 練習問題

ここまでで，一通りピクタグラミングの使い方を見てきました．次は，以下の練習問題に取り組んでみましょう．

練習問題 1

HPML タグの属性を使用して，図 9-1 のような，サッカーを禁止するピクトグラムを作成してください．

図 9-1. サッカー禁止のピクトグラム

練習問題 2

HPML タグに CSS を適用して，図 9-2 のような，バトンパスのピクトグラムを作成してください．

図 9-2. バトンパスのピクトグラム

10. オリジナルの作品制作

HPML を用いることで，いろいろなピクトグラムを作成することができるようになりました．また，HTML と組み合わせることで，ピクトグラムを含んだコンテンツを作成することもできます．図 10-1 のような作品を作ることも可能です．

図 10-1. ピクタグラミングの作品例
（左：段差に注意するように伝えるコンテンツ，右：ピクトグラムの図鑑風コンテンツ）

ピクトグラムを使うと上手く情報を伝えることができそうなものはありますか？ピクタグラミングでピクトグラムを含んだオリジナルのコンテンツを作成してみましょう．