Cours 1 : intro à la prog web et mobile

Web — Applications web et mobile

Christophe Gonzales

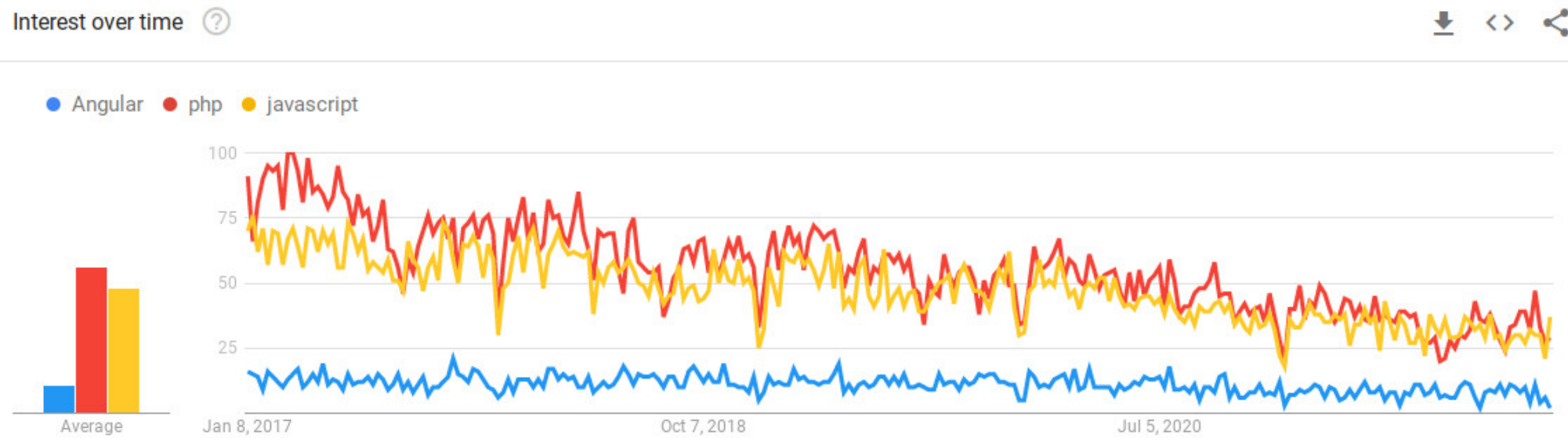
Choix des technologies

Contexte historique du web

1989	World Wide Web / HyperText Markup Language (HTML)
1994	Personal Home Page Tools (PHP)
1995	Javascript
1998	asynchronous JavaScript and XML (Ajax)
2000	ASP.NET
2009	NodeJS (node)
2010	AngularJS
2012	TypeScript
2013	React
2014	Angular, Vue

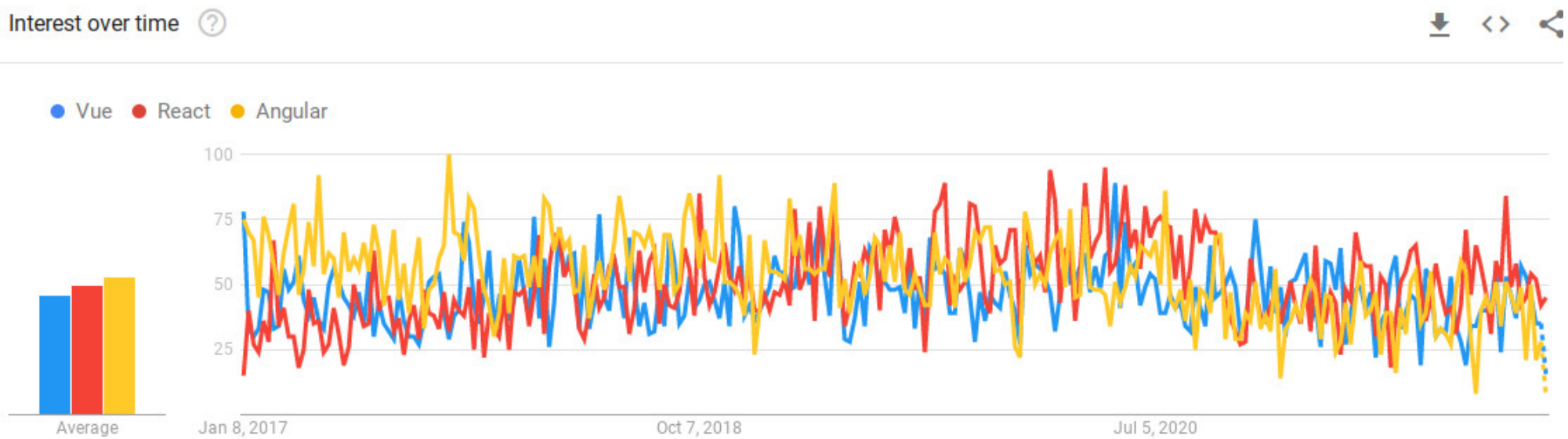
Tendances (1/2)

- Tendances de requêtes google sur Angular/PHP/Javascript sur les 5 dernières années (source : Google trends) :



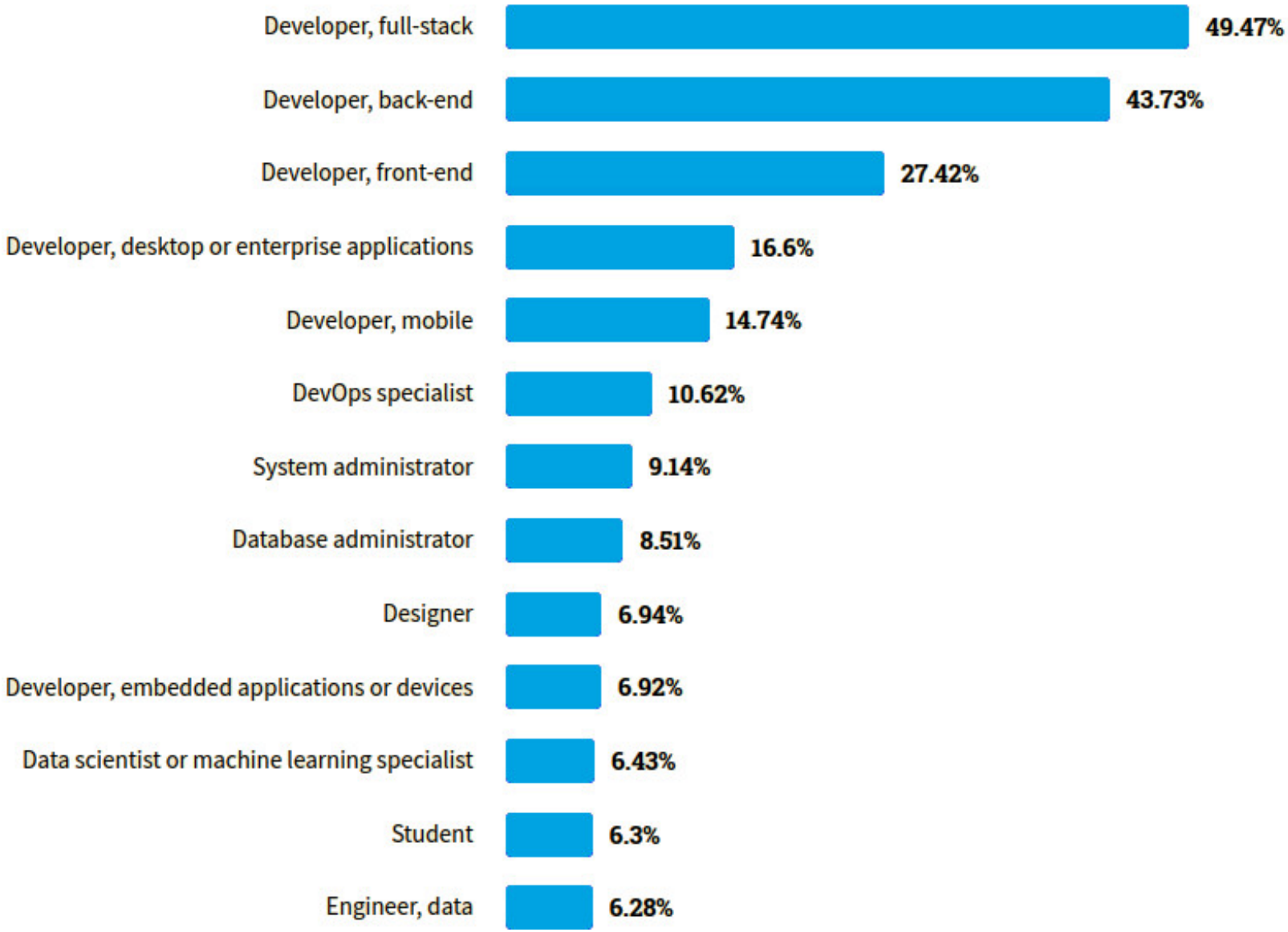
Tendances (2/2)

- Tendances de requêtes google sur Vue/React/Angular sur les 5 dernières années (source : Google trends) :



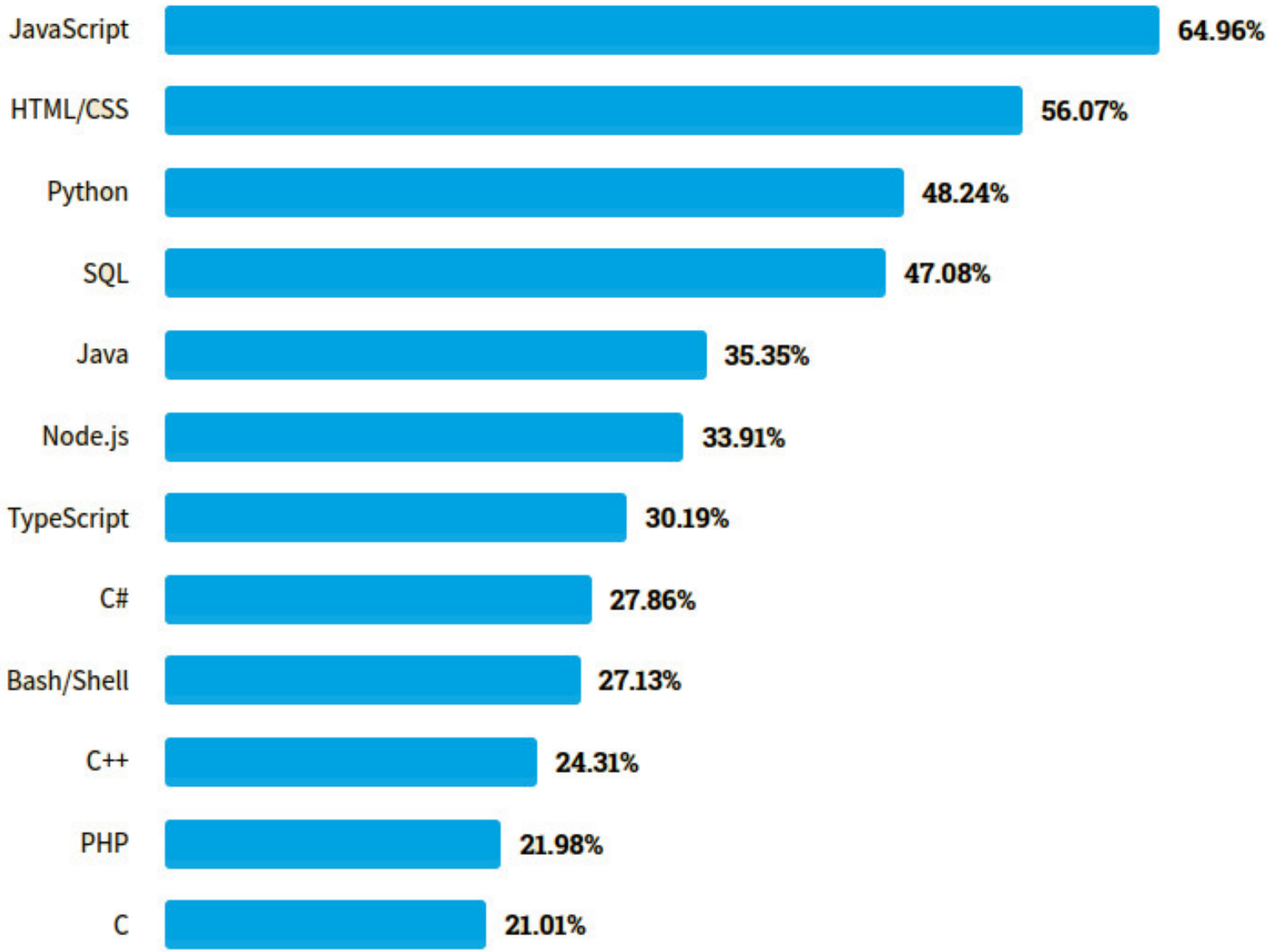
Types de développeurs

- Étude de StackOverflow de 2021 :

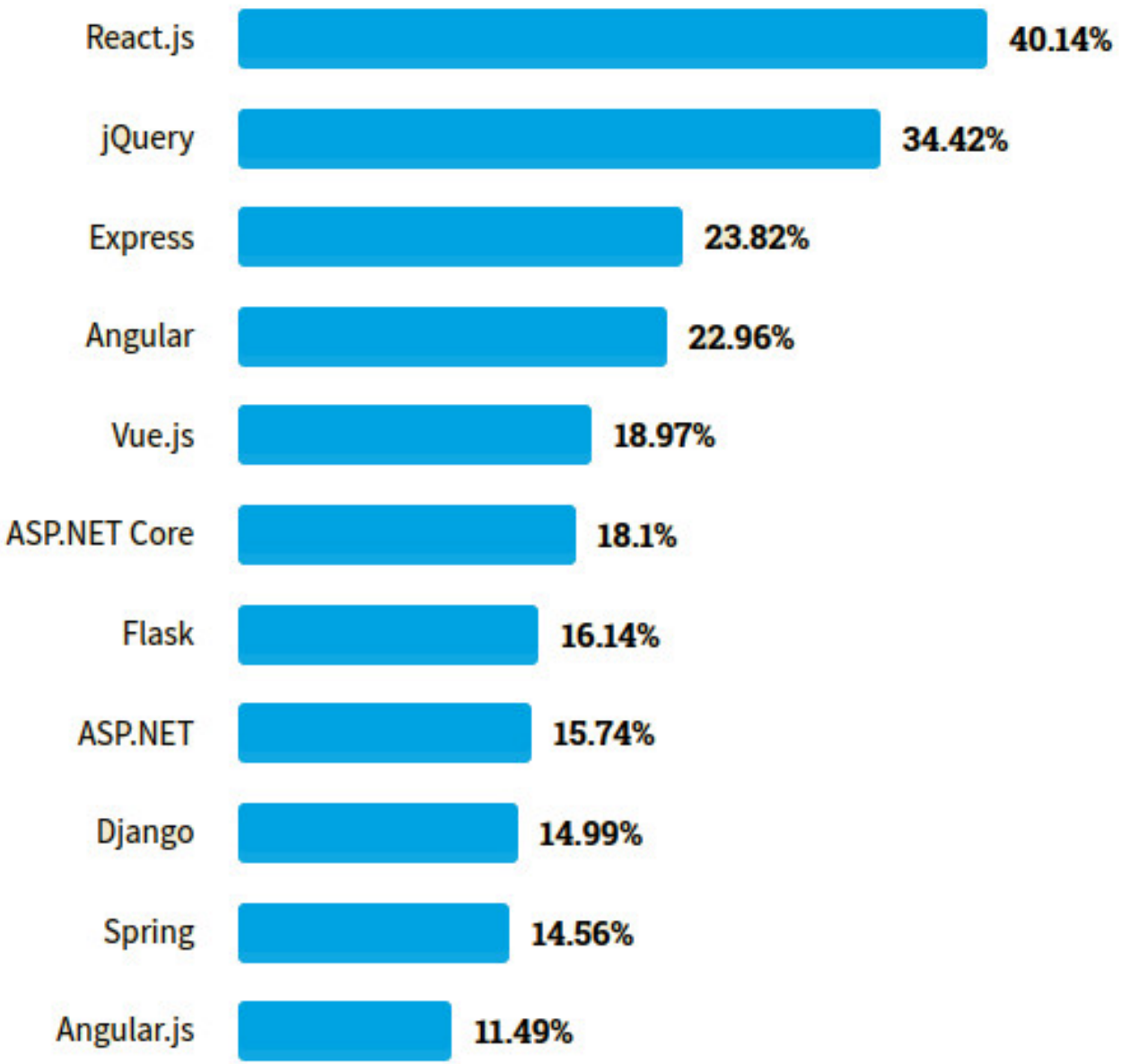


Langages de programmation utilisés

- Étude de StackOverflow de 2021 :



- Étude de StackOverflow de 2021 :



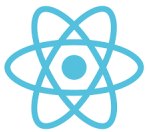
Quelle technologie connaître/choisir ?

: WordPress, Drupal, Joomla
80% des sites web dans le monde


: Facebook, Twitter, Netflix, Instagram, Airbnb, Yahoo Mail


: Paypal, GMail, Netflix, Lego, Weather, Delta

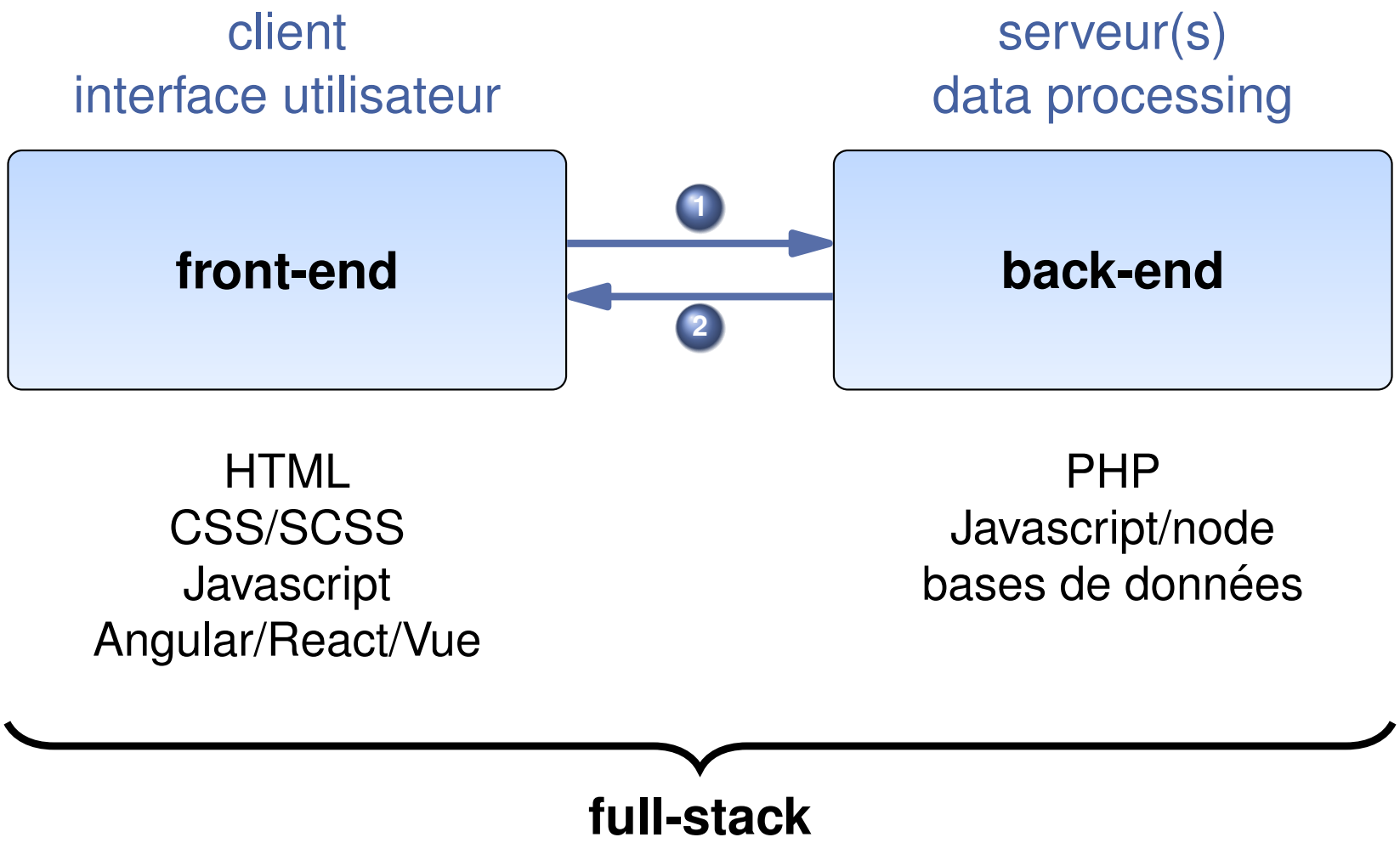

: Nintendo, Gitlab, Alibaba, Xiaomi


stable


update majeure
tous les 6 mois


API 90% similaire
entre la V1 et la V2

Un peu d'architecture...



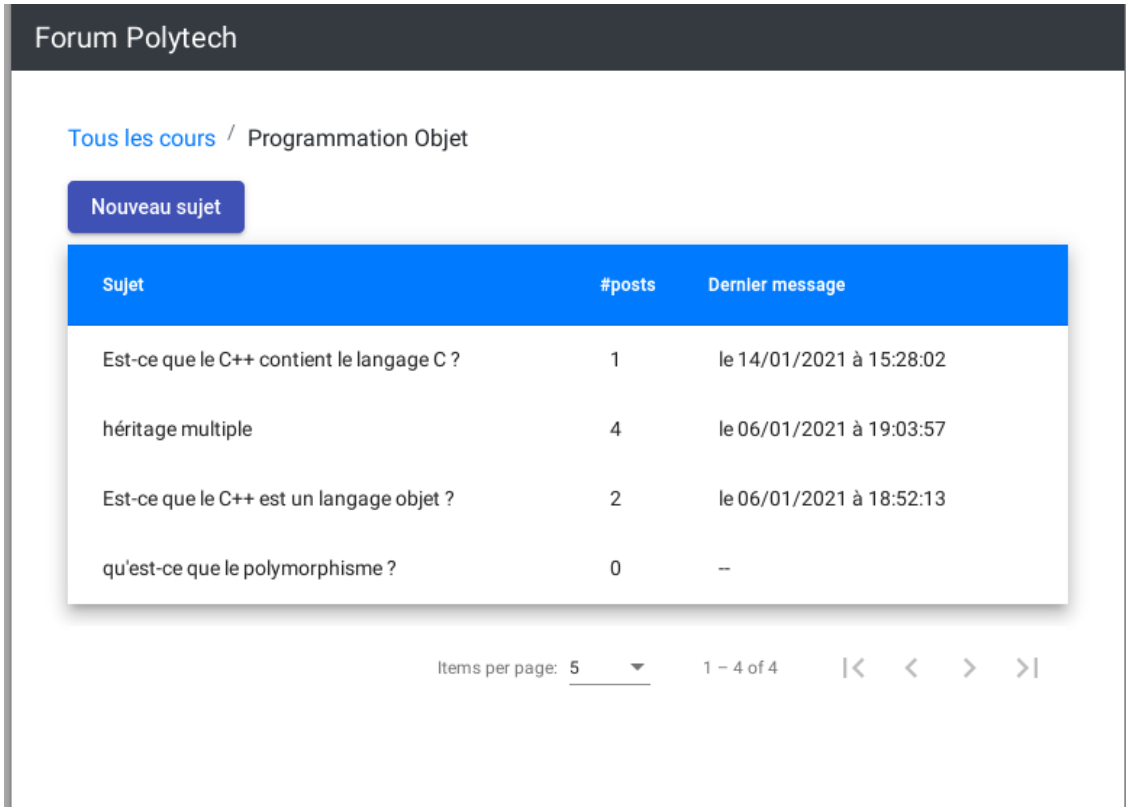
Plan du module

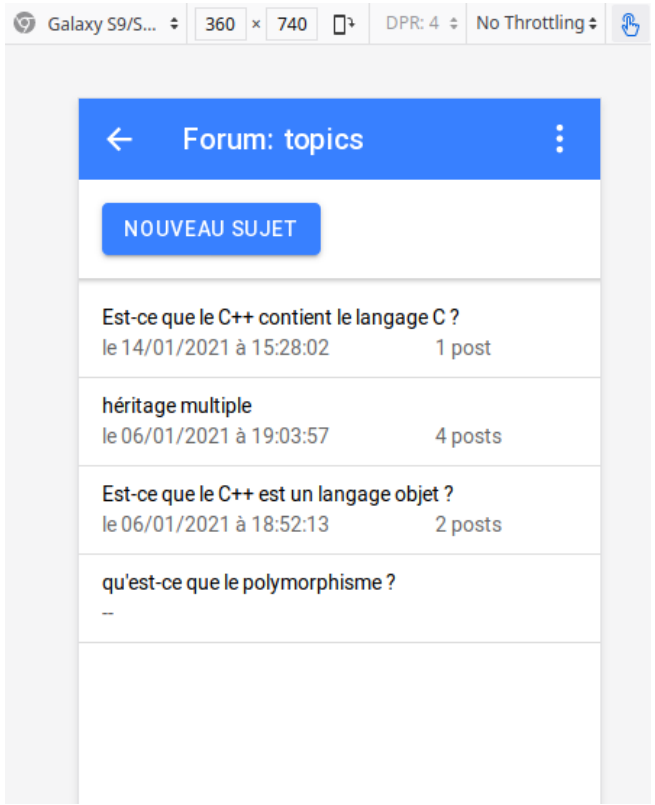
Objectif : développement full-stack / mobile

Plan du module :

1. HTML/CSS/PHP/Javascript
2. Angular/TypeScript
3. Node/Express
4. Ionic

Objectif des TPs





- `https://christophe-gonzales.pedaweb.univ-amu.fr/forum`
- `https://christophe-gonzales.pedaweb.univ-amu.fr/forum-ionic`

Technologies : PHP, Angular, node/express, ionic

Rappels HTML/CSS

Rappels élémentaires de HTML


```
<!doctype html>
<html class="no-js" lang="">

<head>
  <meta charset="utf-8">
  <title></title>
  <meta name="description" content="">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="css/main.css">
</head>

<body>
  <p>Hello world!</p>

  <script src="js/vendor/modernizr-3.8.0.min.js"></script>
  <script src="https://code.jquery.com/jquery-3.4.1.min.js" integrity="sha256-CSXorXvZcTkaix6Yvo6HppcZGetbYMG
  <script>window.jQuery || document.write('<script src="js/vendor/jquery-3.4.1.min.js"><\/script>')</script>
  <script src="js/plugins.js"></script>
  <script src="js/main.js"></script>

  <!-- Google Analytics: change UA-XXXXX-Y to be your site's ID. -->
  <script>
    window.ga = function () { ga.q.push(arguments) }; ga.q = []; ga.l = +new Date;
    ga('create', 'UA-XXXXX-Y', 'auto'); ga('set','transport','beacon'); ga('send', 'pageview')
  </script>
  <script src="https://www.google-analytics.com/analytics.js" async></script>
</body>

</html>
```

Balises HTML

Balises

- En général :

  `<balise>...</balise>` ou

  `<balise attribut1="val1" attribut2="val2">...</balise>`

- Quelques exceptions :

  `<balise/>` ou

  `<balise attribut1="val1" attribut2="val2" />`

- ***Exemples :***

Balise	Signification
`<p></p>`	paragraphe
`<div></div>`	conteneur de « division »
`<span></span>`	conteneur « inline »
`<ol></ol>, <ul></ul>, <li></li>`	listes
`<img src="..." />`	images
` `	retour à la ligne

Attributs particuliers

Attributs

- **`class`** : classe $\Longrightarrow$ permet de manipuler l'élément via CSS ou Javascript (sélecteur)
- **`id`** : identifiant unique $\Longrightarrow$ permet de manipuler l'élément via CSS ou Javascript (sélecteur)

- ***Exemple :***

HTML

```
<h1>Section 1 :</h1>
<div class="mydiv">
Hello
<span>promo</span>
<span id="myspan">3A</span>
</div>
```

CSS

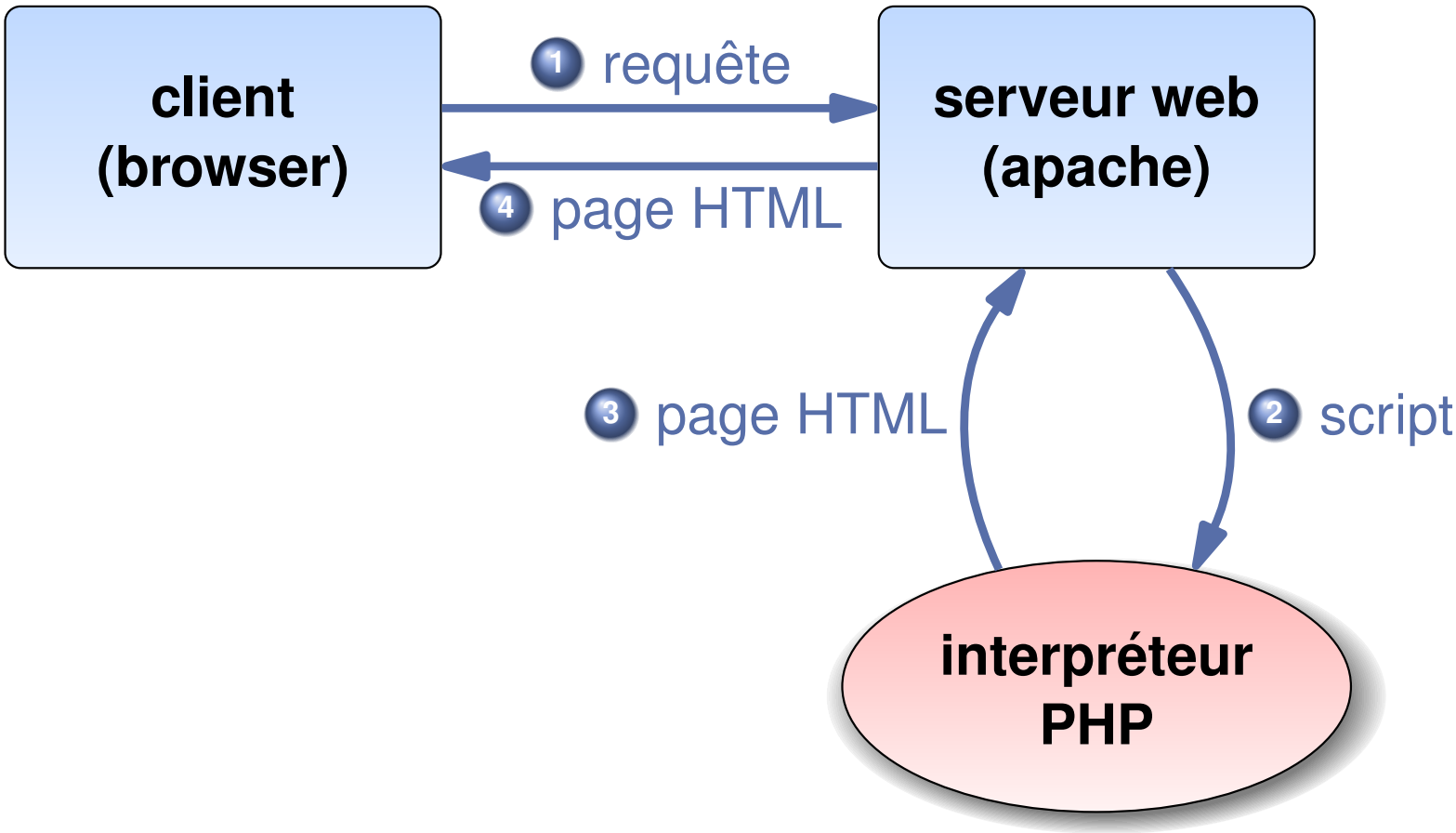
```
div.mydiv { text-align: right; }
div span { color: red; }
#myspan {
  color: blue;
  font-weight: bold;
}
```

Section 1 :

Hello promo 3A

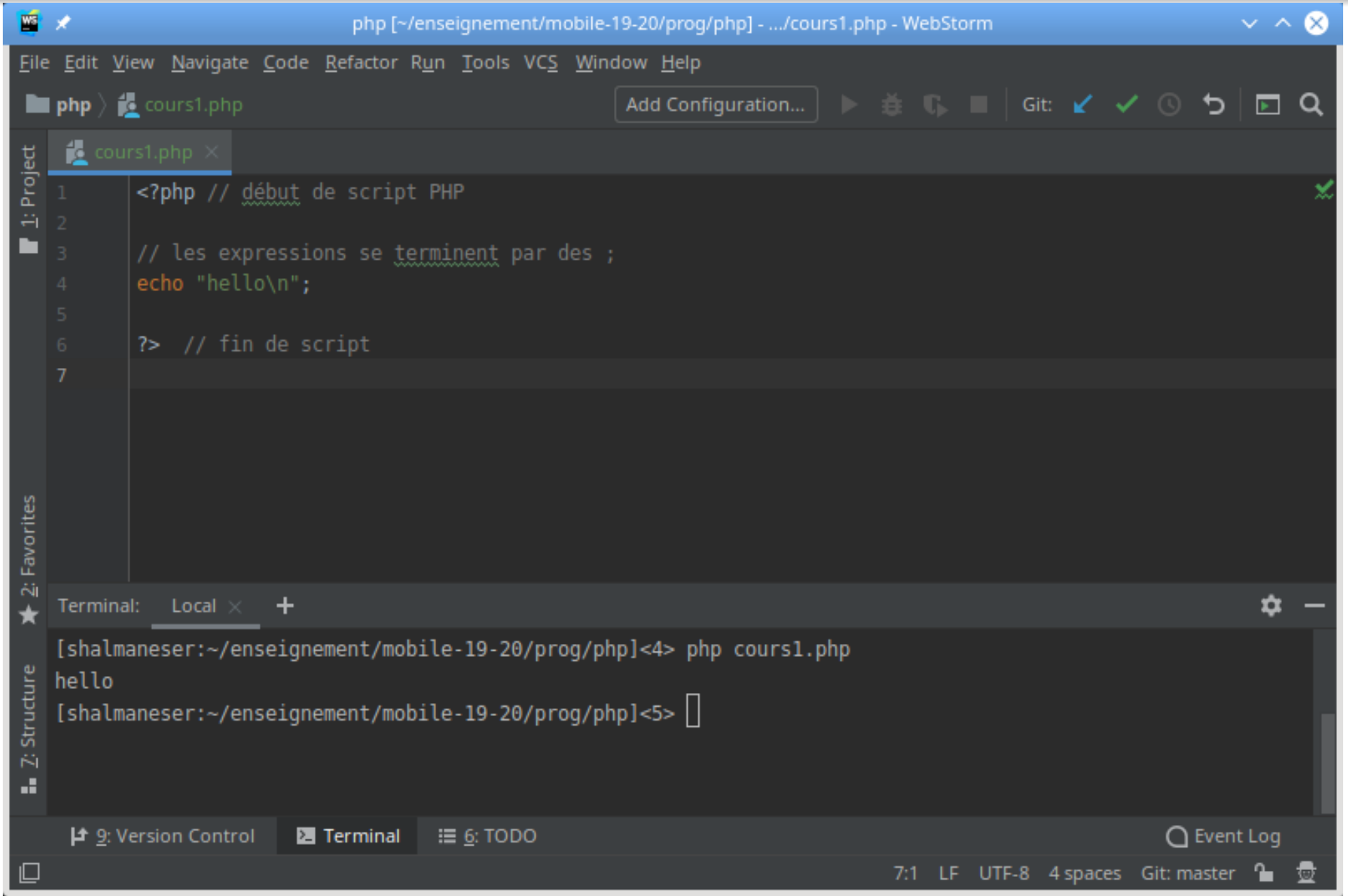
Introduction à PHP

Architecture autour de PHP

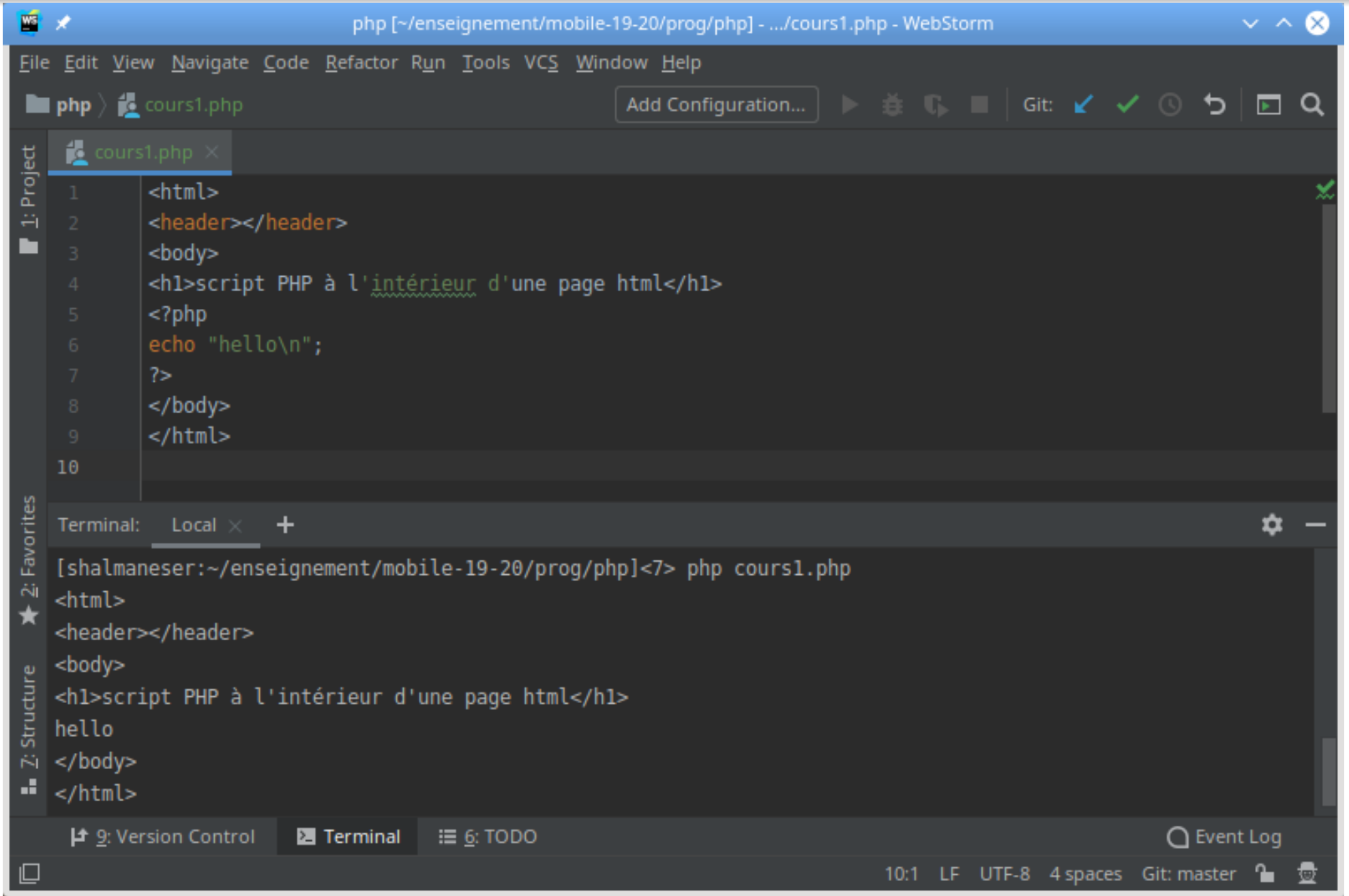


$\Longrightarrow$ But : les scripts PHP produisent du texte HTML !
ils sont aussi utiles pour produire des données JSON

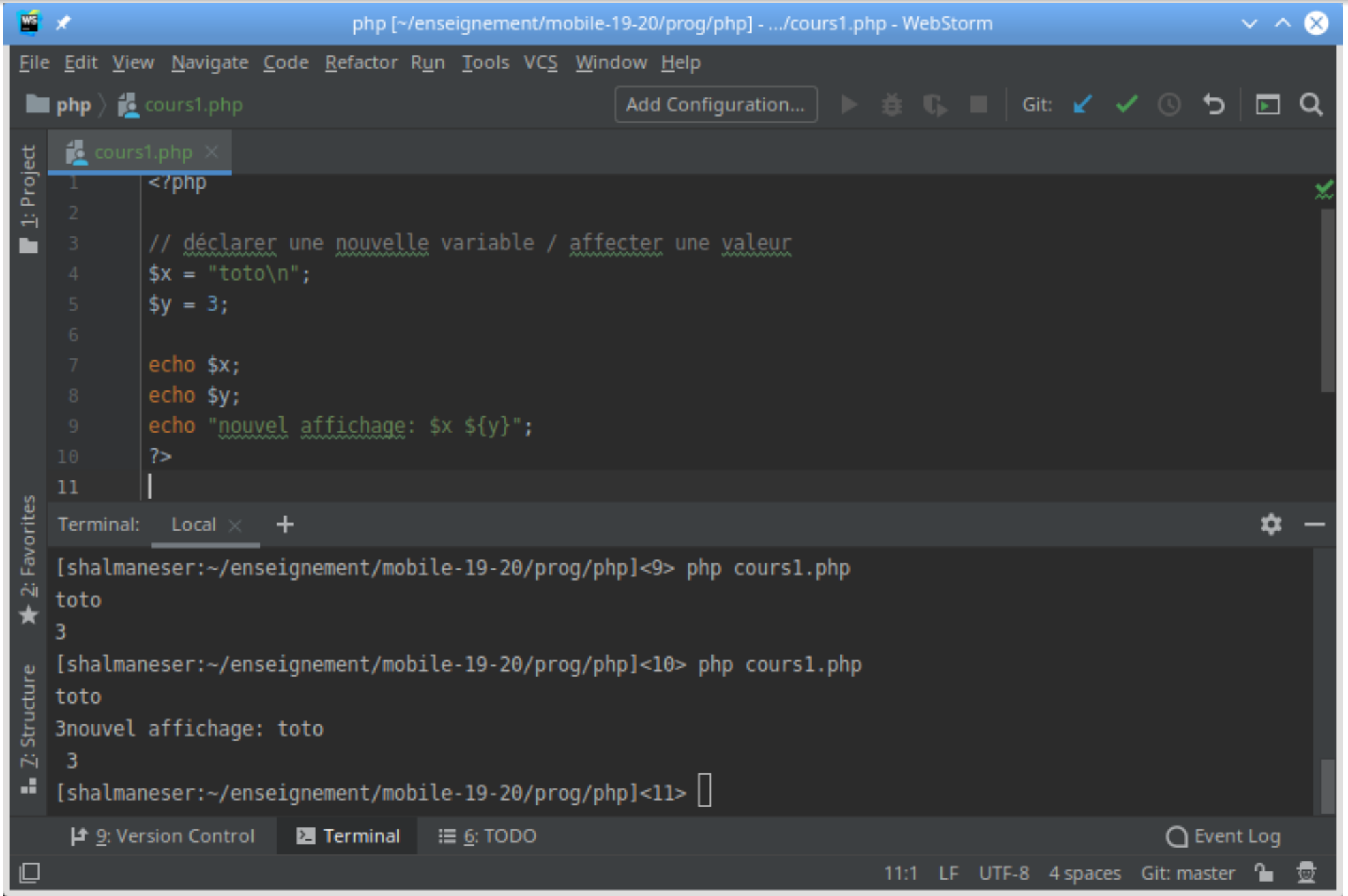
Scripts PHP


php [~/enseignement/mobile-19-20/prog/php] - .../cours1.php - WebStorm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
php cours1.php
Add Configuration...
Git:
cours1.php
1: Project
2: Favorites
7: Structure
1 <?php // début de script PHP
2
3 // les expressions se terminent par des ;
4 echo "hello\n";
5
6 ?> // fin de script
7
Terminal: Local
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<4> php cours1.php
hello
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<5>
9: Version Control
Terminal
6: TODO
Event Log
7:1 LF UTF-8 4 spaces Git: master

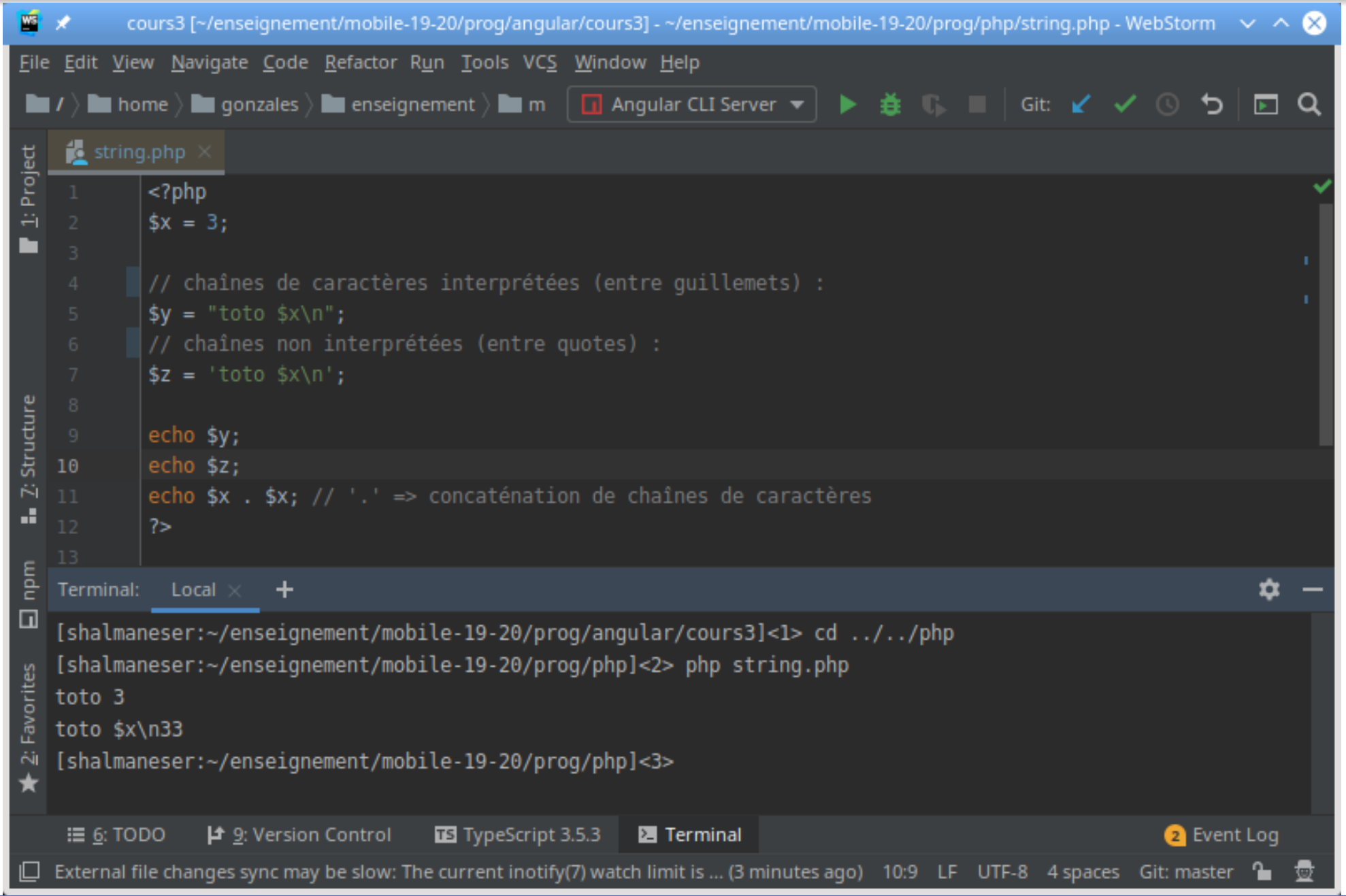
PHP à l'intérieur de balises HTML



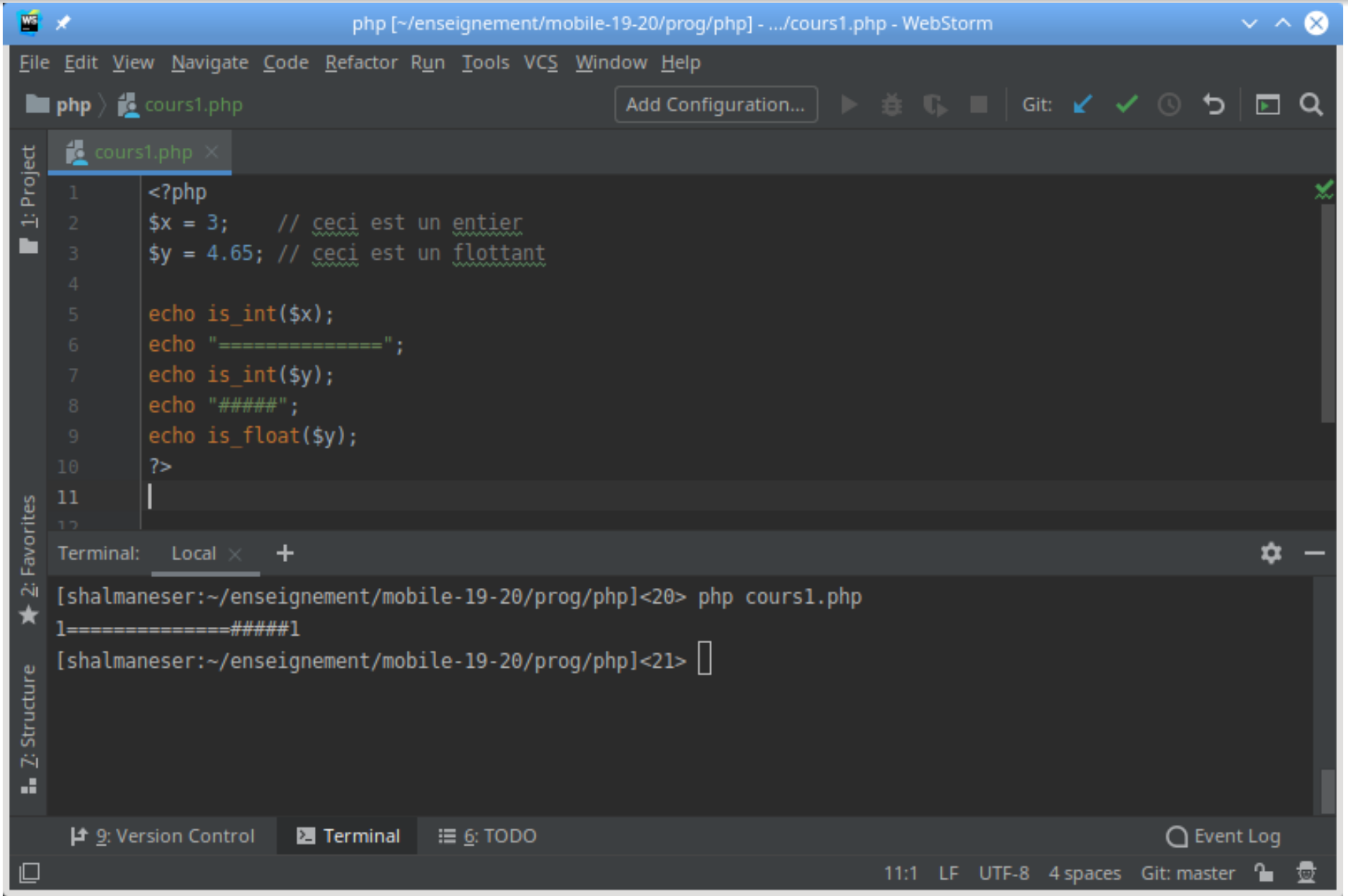
Déclarer des variables/affecter des valeurs



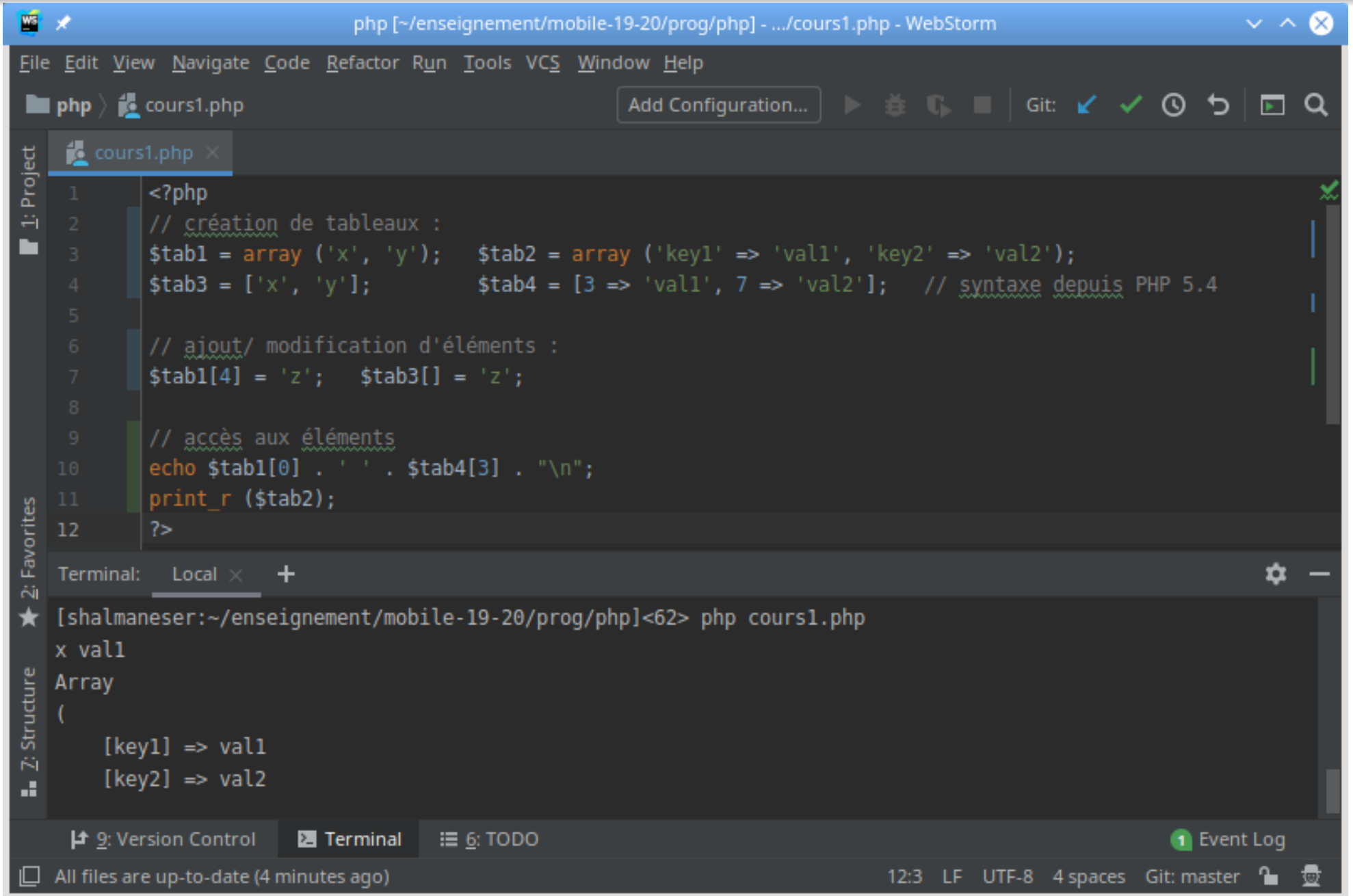
Les chaînes de caractères



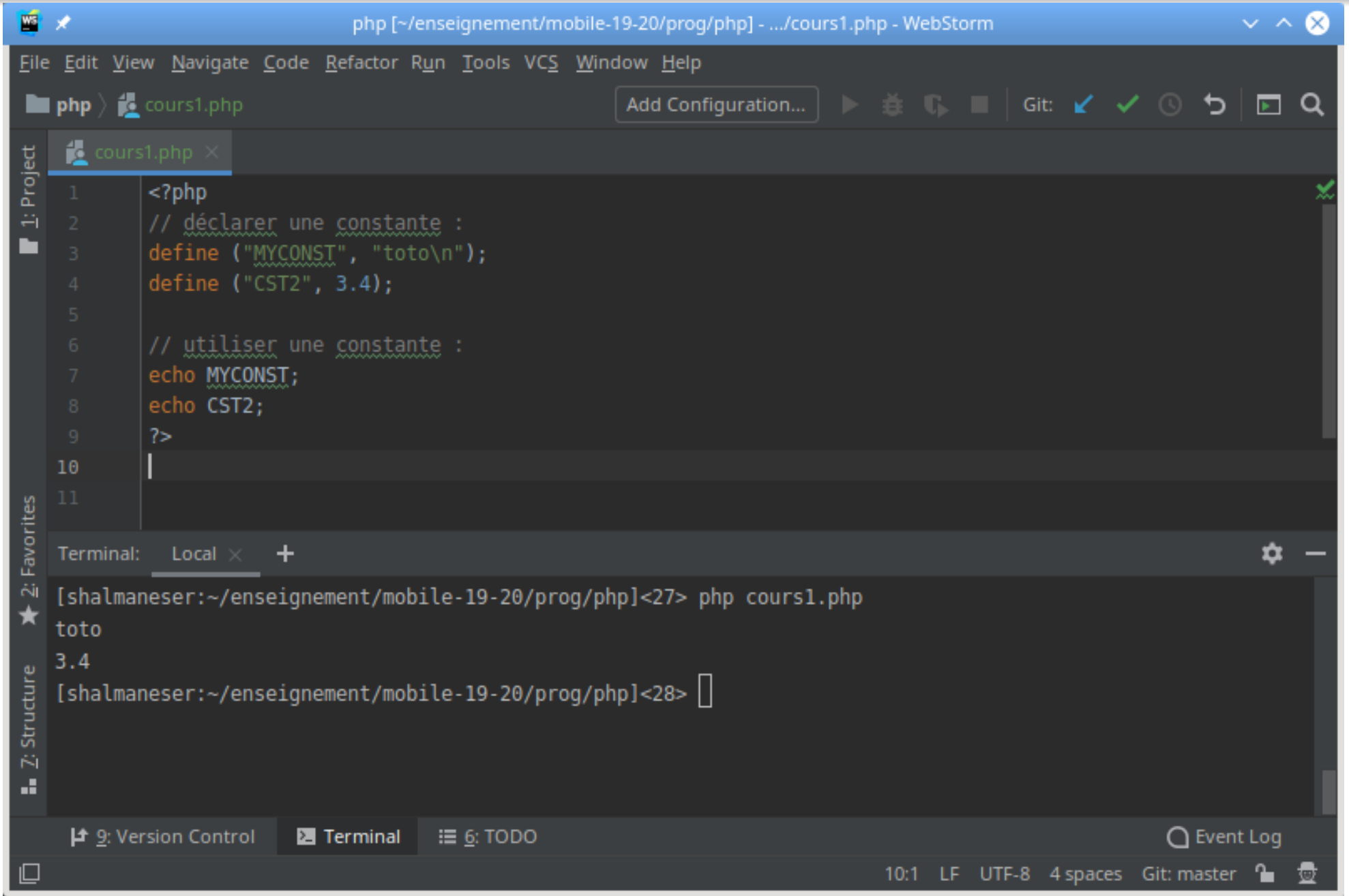
Les nombres


php [~/enseignement/mobile-19-20/prog/php] - .../cours1.php - WebStorm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
php cours1.php
Add Configuration...
Git:
cours1.php
1: Project
2: Favorites
7: Structure
1 <?php
2 $x = 3; // ceci est un entier
3 $y = 4.65; // ceci est un flottant
4
5 echo is_int($x);
6 echo "==========";
7 echo is_int($y);
8 echo "#####";
9 echo is_float($y);
10 ?>
11
Terminal: Local
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<20> php cours1.php
1==========#####1
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<21>
9: Version Control
Terminal
6: TODO
Event Log
11:1 LF UTF-8 4 spaces Git: master

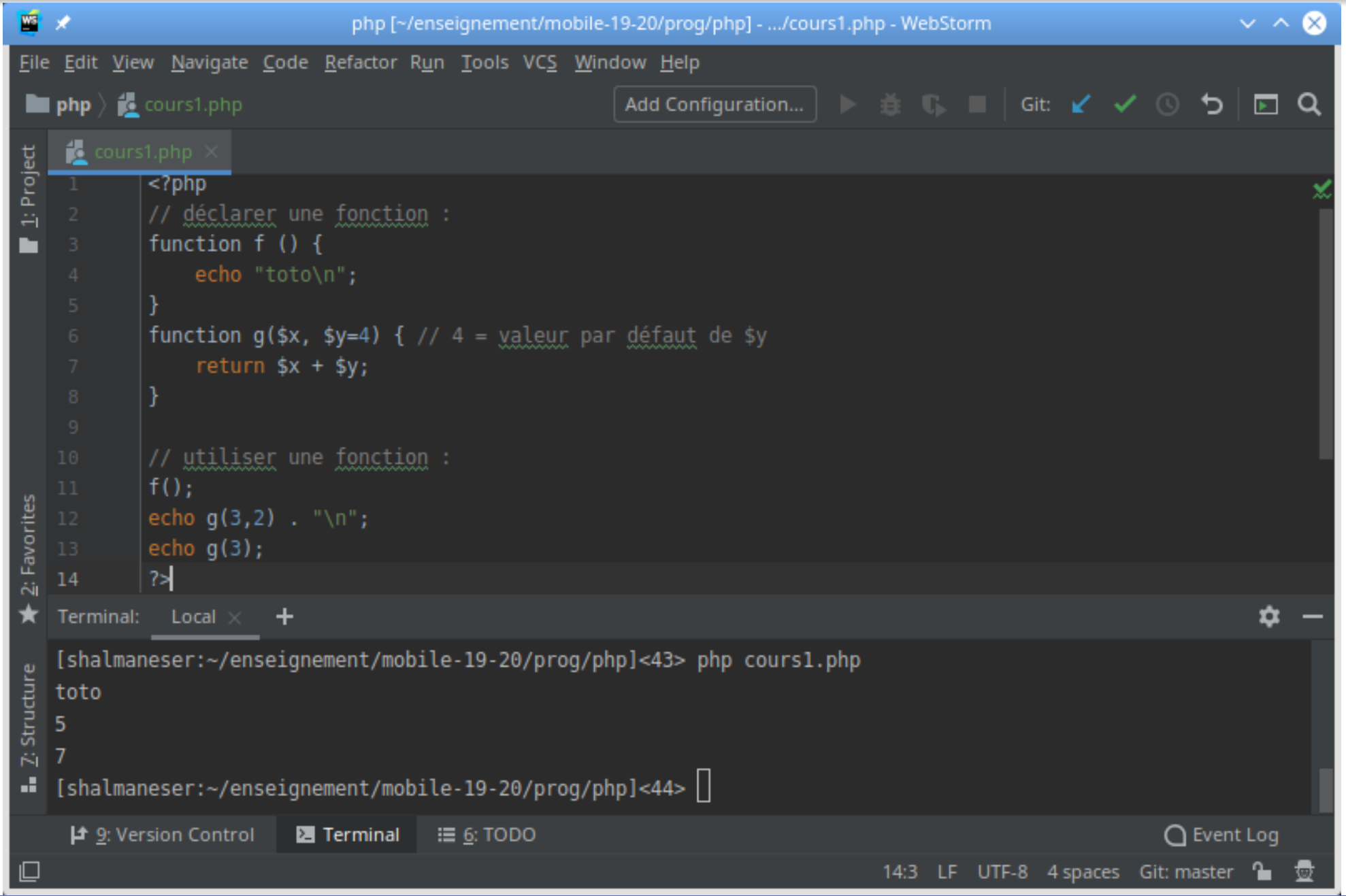
Les tableaux



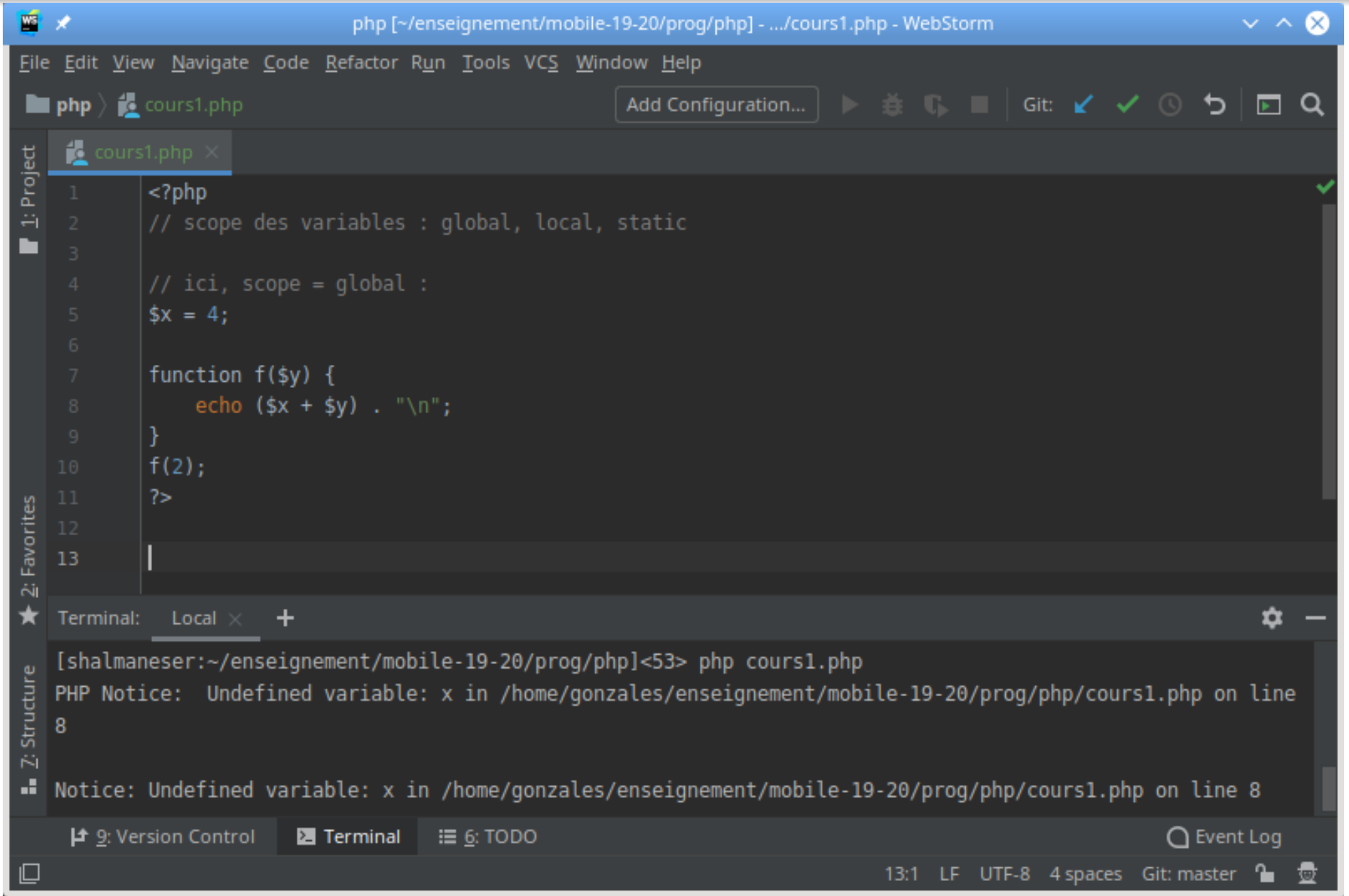
Les constantes



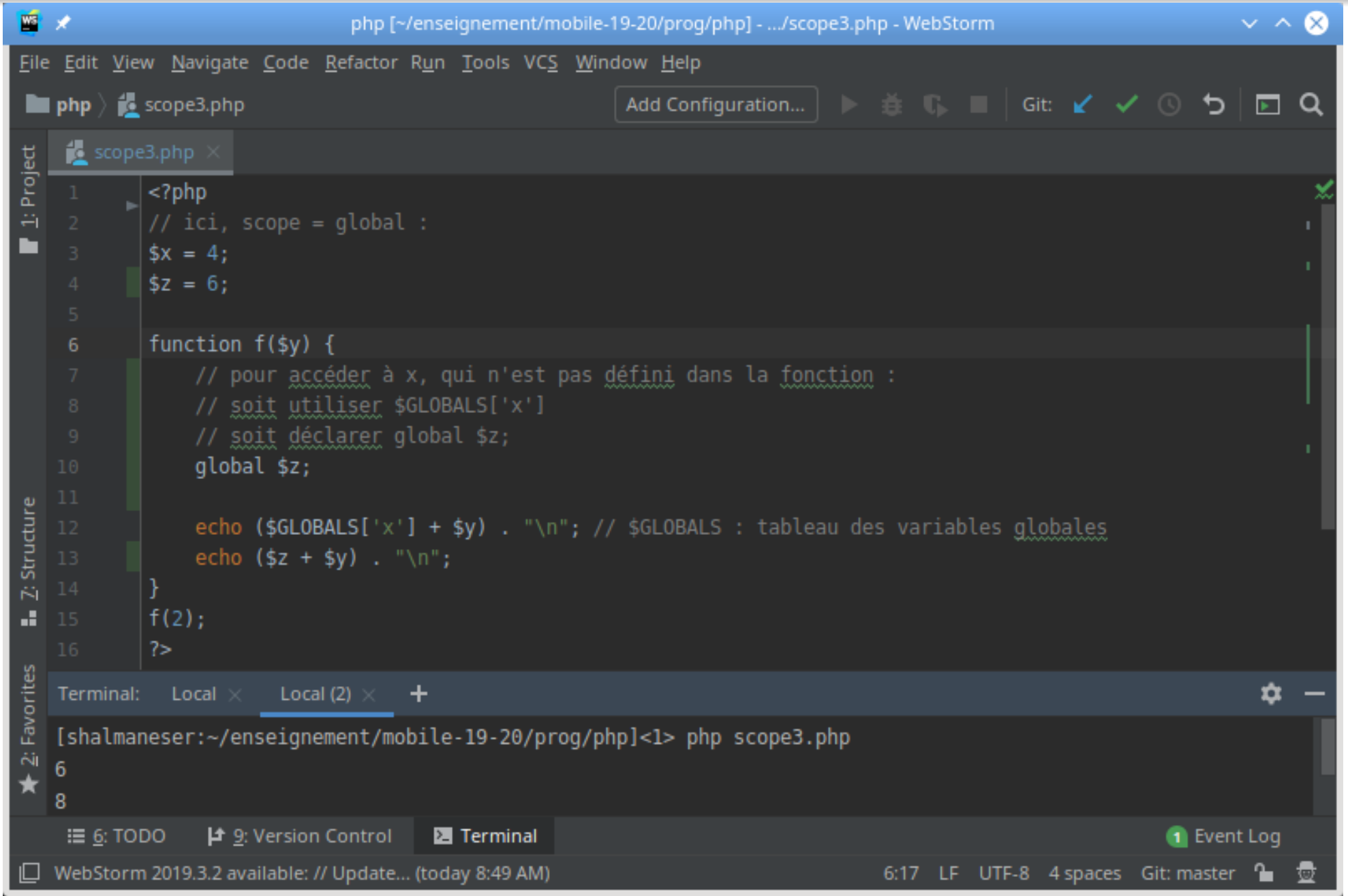
Les fonctions



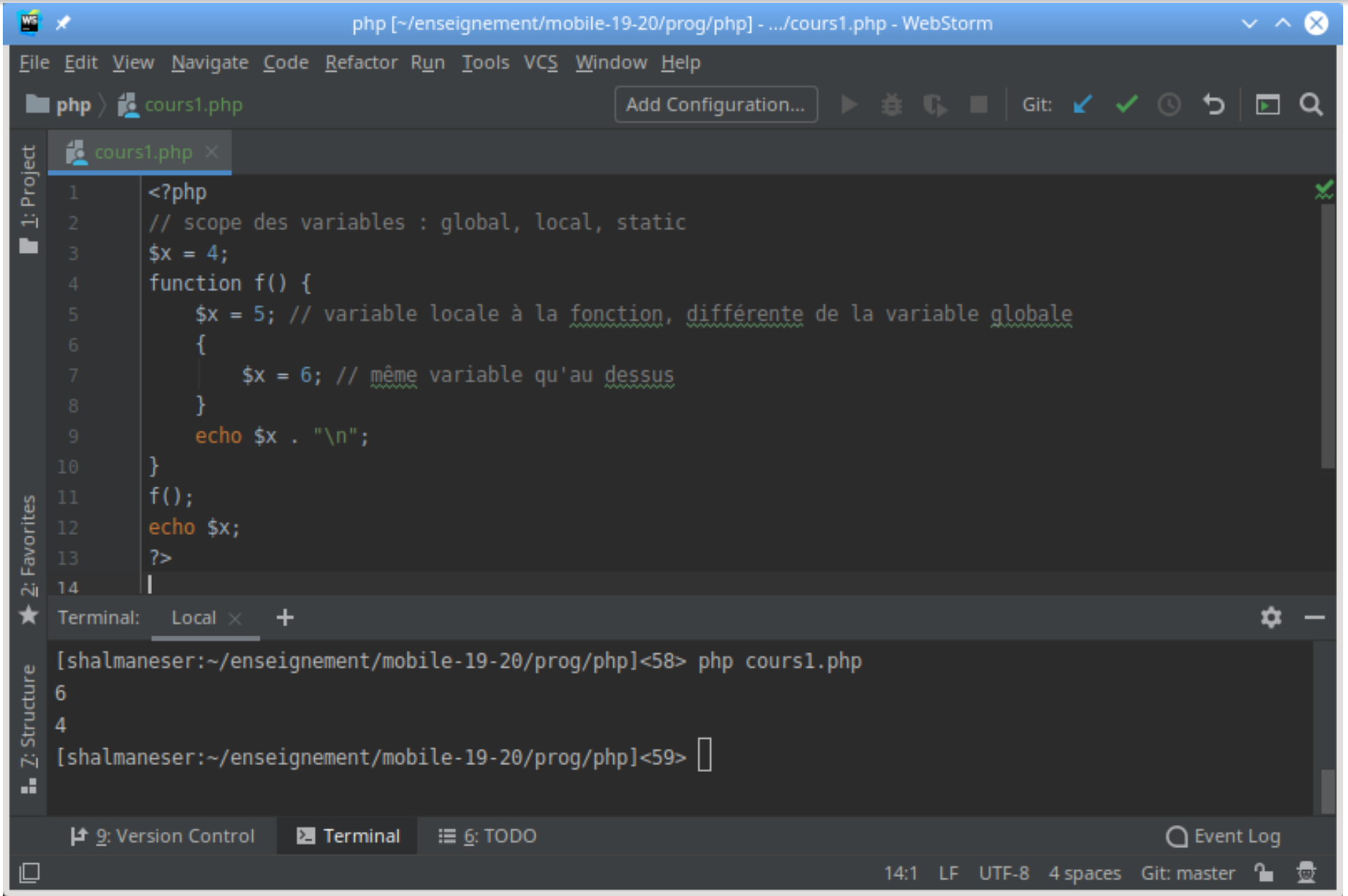
La portée des variables (1/4)



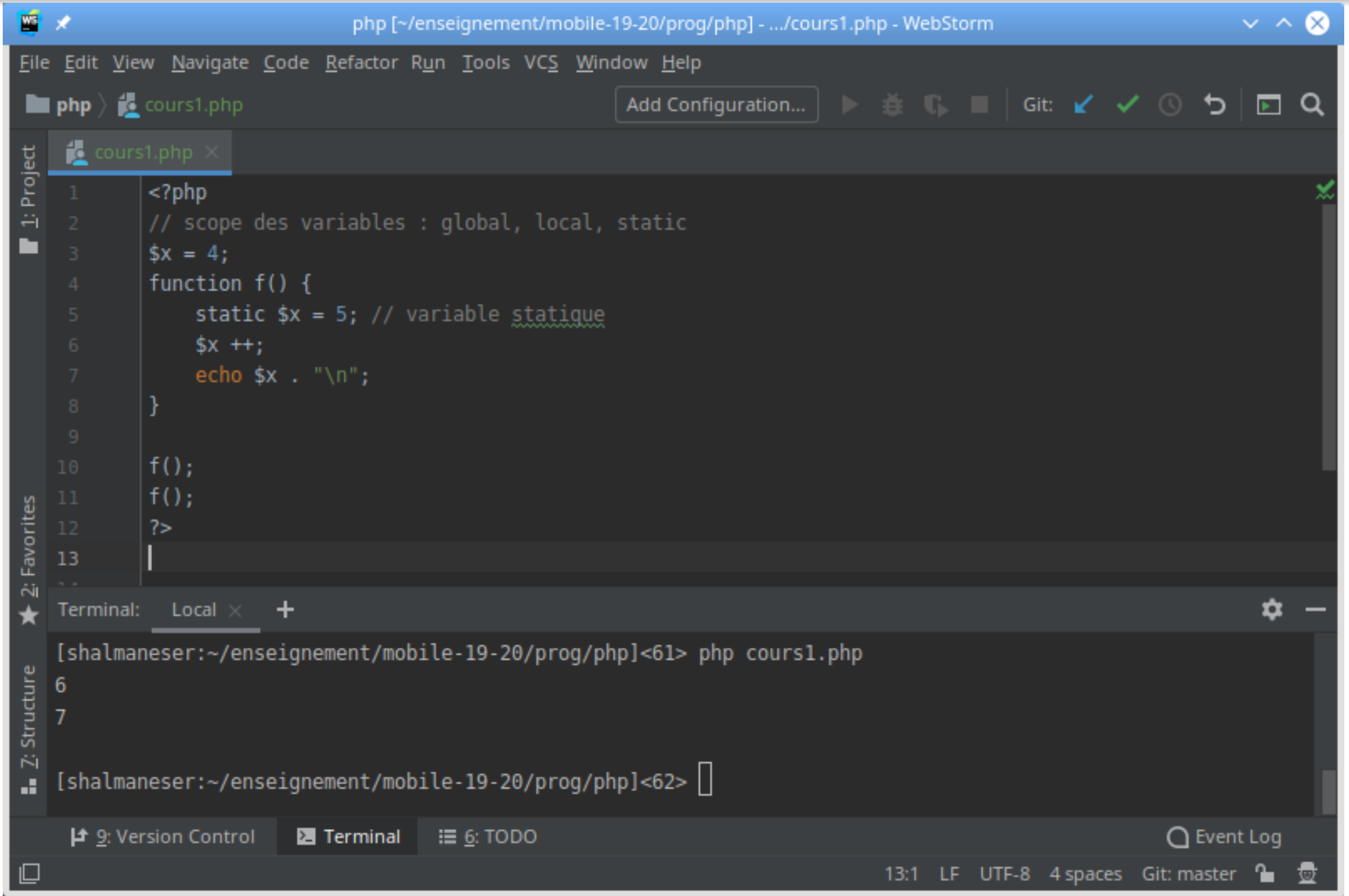
La portée des variables (2/4)



La portée des variables (3/4)



La portée des variables (4/4)



Les variables superglobales

- **Superglobal :** accessible dans tous les scopes
- Ce sont des tableaux (`$GLOBALS['x']`)
- Variables contenant les infos transmises par l'utilisateur au serveur web :
  - `$_POST`
  - `$_GET`
  - `$_REQUEST` = `$_POST` + `$_GET`
- Variables contenant les infos sur le serveur web :
  - `$_SERVER`
- Variables contenant les infos sur les fichiers uploadés :
  - `$_FILES`
- Variables de session et cookies :
  - `$_SESSION`
  - `$_COOKIE`

Forms HTML et variables superglobales PHP

php [~/enseignement/mobile-19-20/prog/php] - .../form.html - WebStorm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

php | Add Configuration... | Git:

1: Project

form.html

```
1  <!doctype html>
2  <html>
3  <body>
4    <form method="post" action="form.php" id="act">
5      <input type="text" name="texte" width="30" />
6      <input type="submit" name="xsub" value="bouton">
7    </form>
8  </body>
9  </html>
```

7: Structure

2: Favorites

form.php

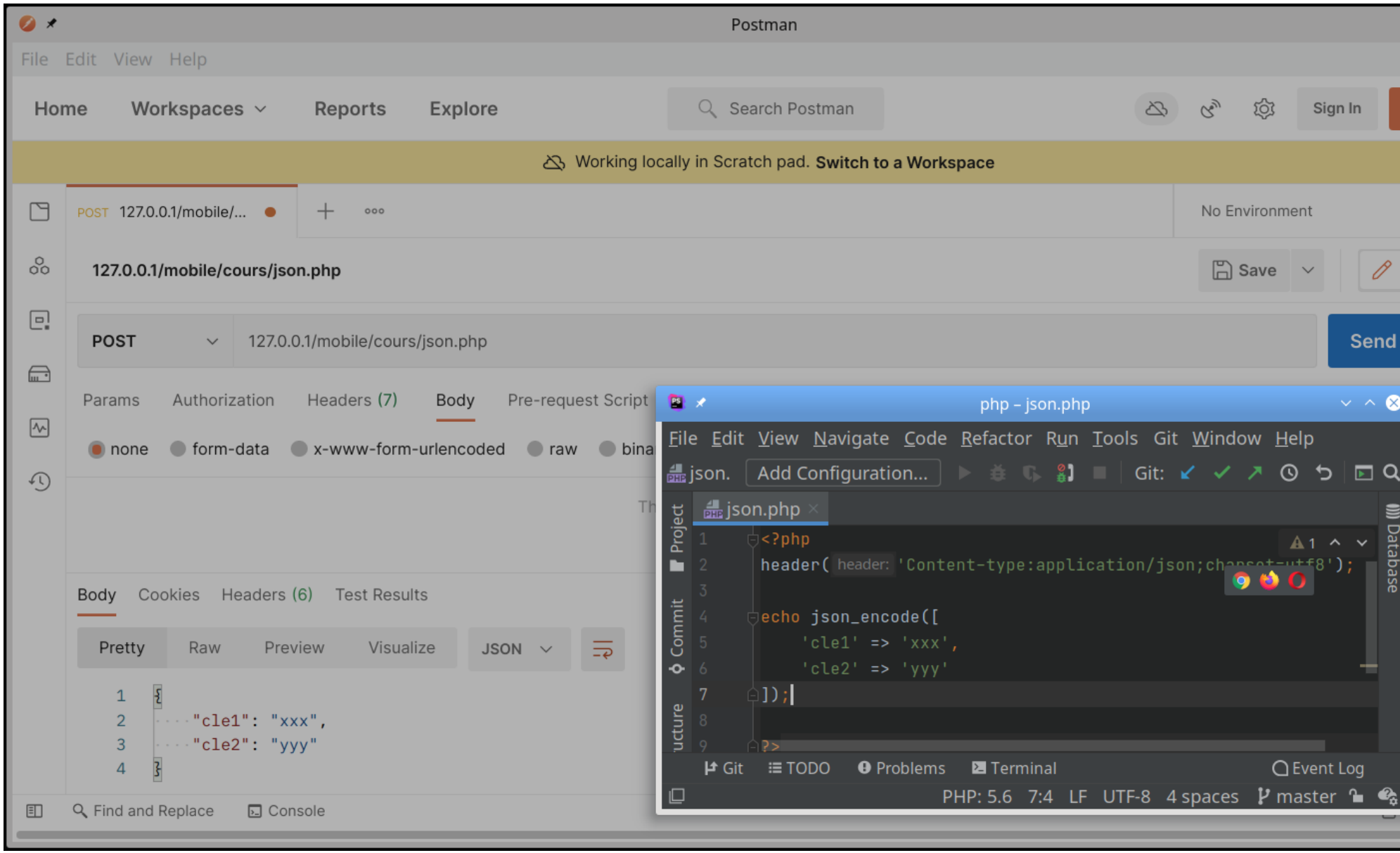
```
1  <?php
2
3  echo '<pre>';
4
5  print_r ($_REQUEST);
6
7  echo '</pre>';
8
9  ?>
```

6: TODO | 9: Version Control | Terminal | Event Log

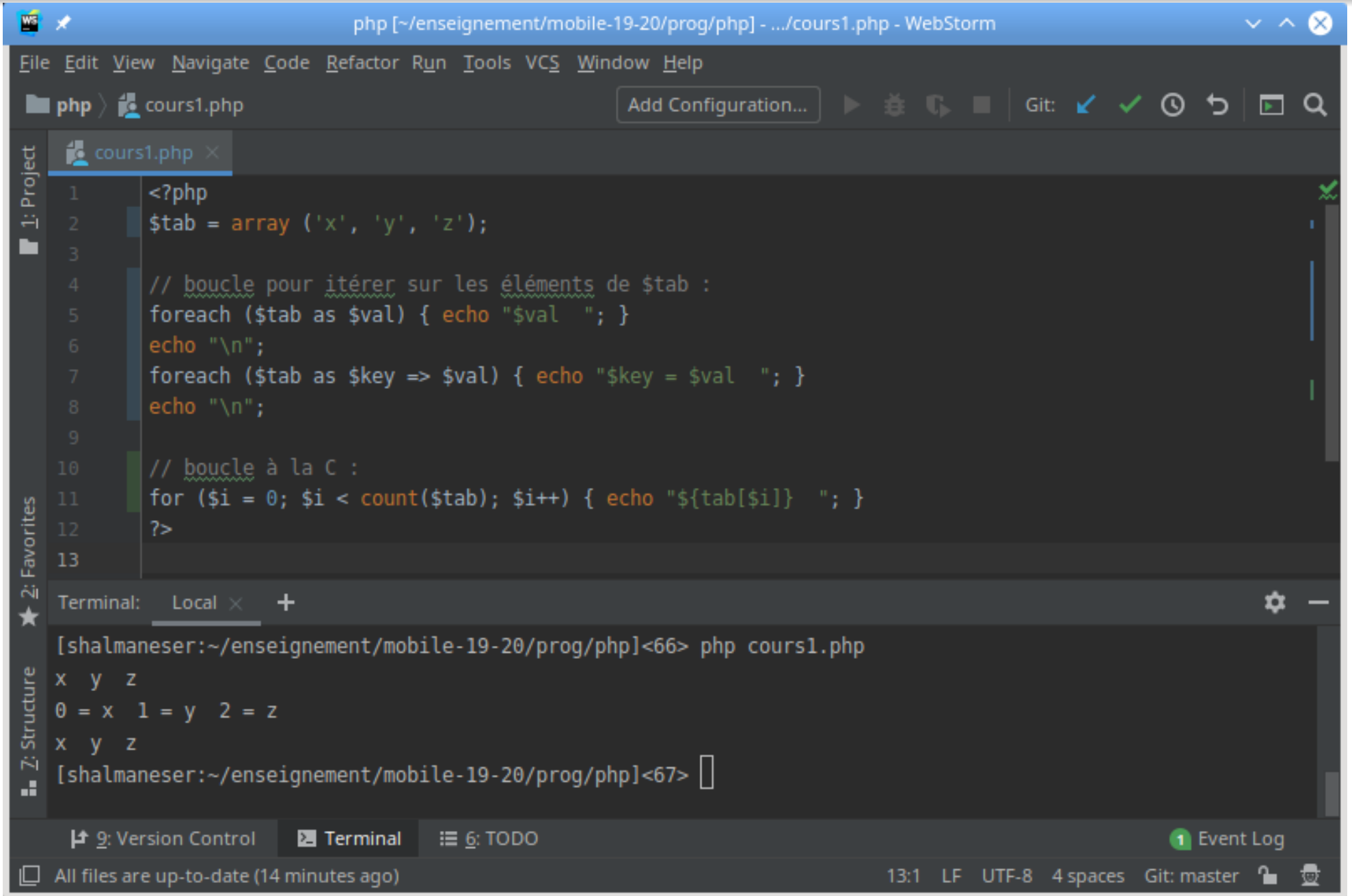
1:1 LF UTF-8 2 spaces* Git: master


```
Array
(
    [texte] => sss
    [xsub] => bouton
)
```

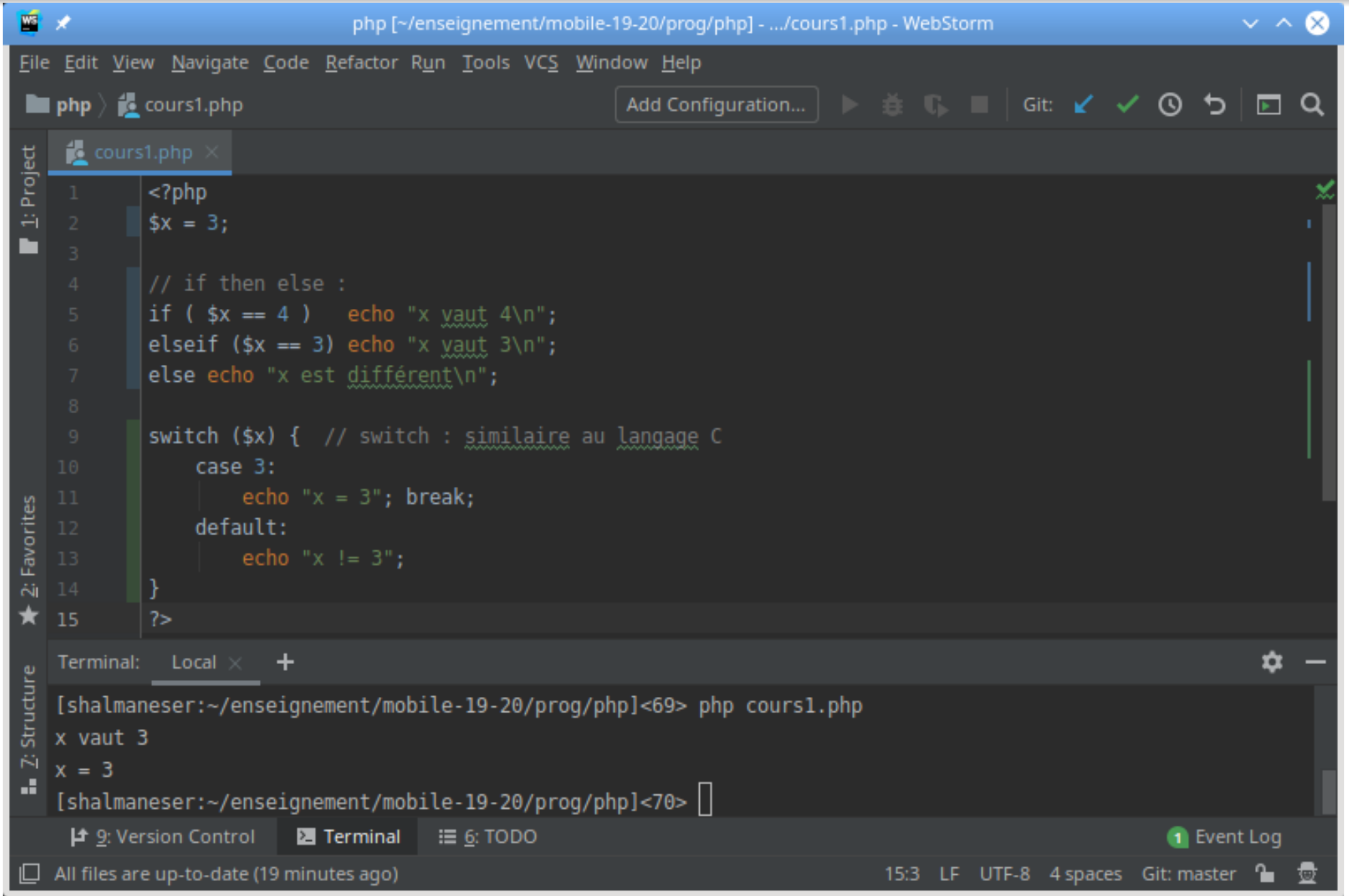
Backend PHP et transfert d'informations en JSON



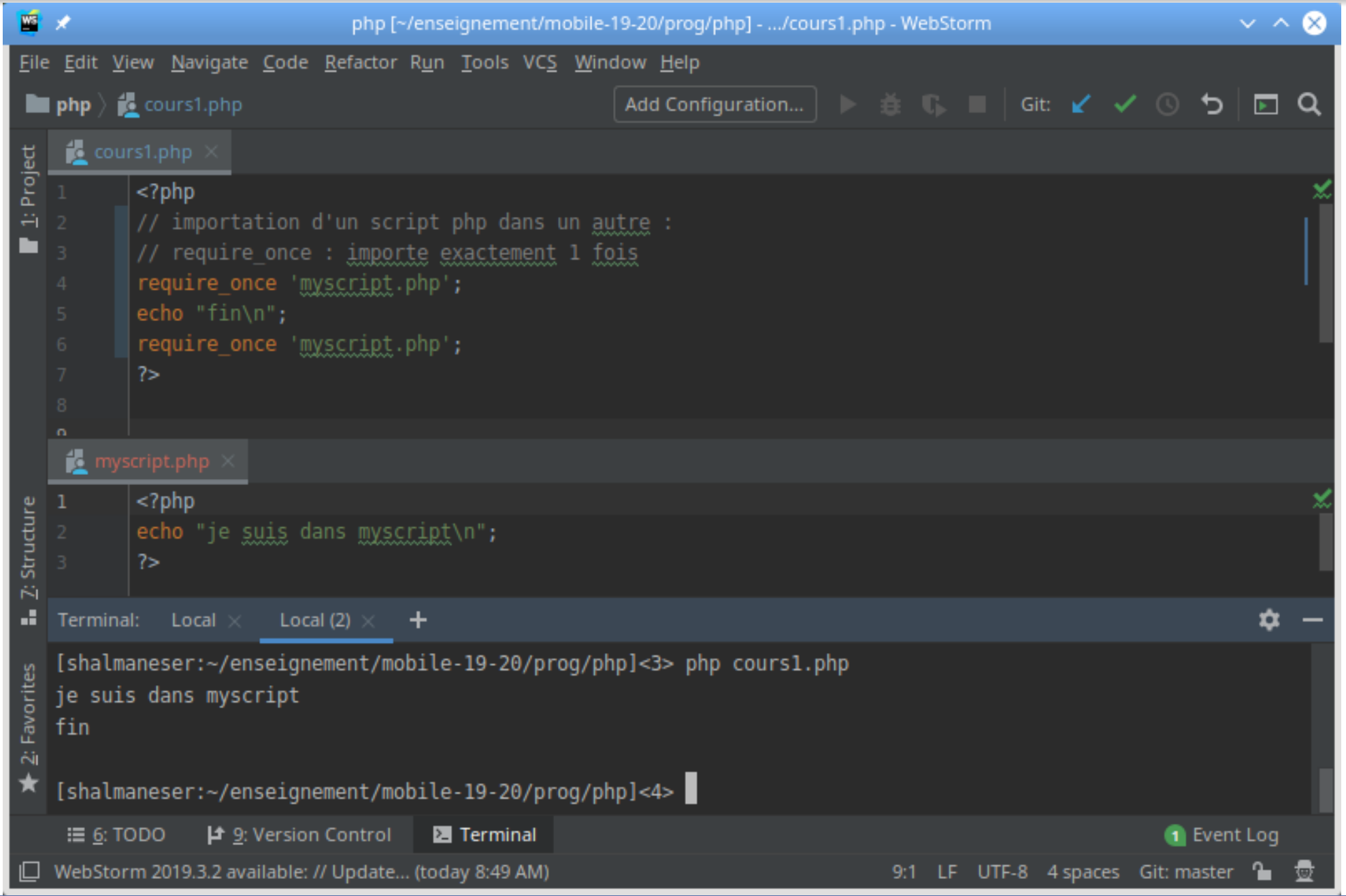
Les boucles


php [~/enseignement/mobile-19-20/prog/php] - .../cours1.php - WebStorm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
php cours1.php
Add Configuration...
Git:
cours1.php
1: Project
2: Favorites
7: Structure
1 <?php
2 $tab = array ('x', 'y', 'z');
3
4 // boucle pour itérer sur les éléments de $tab :
5 foreach ($tab as $val) { echo "$val "; }
6 echo "\n";
7 foreach ($tab as $key => $val) { echo "$key = $val "; }
8 echo "\n";
9
10 // boucle à la C :
11 for ($i = 0; $i < count($tab); $i++) { echo "${tab[$i]} "; }
12 ?>
13
Terminal: Local
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<66> php cours1.php
x y z
0 = x 1 = y 2 = z
x y z
[shalmaneser:~/enseignement/mobile-19-20/prog/php]<67>
9: Version Control
Terminal
6: TODO
1 Event Log
All files are up-to-date (14 minutes ago)
13:1 LF UTF-8 4 spaces Git: master

Les alternatives



Les importations



Requêtes MySQL $\Longrightarrow$ utiliser un « *database abstraction layer* »

- ici : utilisation de PDO (PHP Data Objects)
- **Avantages :**
  - sécurité (« *prepared statements* »)
  - facilité d'utilisation (« *helpers* »)
  - réutilisabilité (« *API unifiée de bases de données* »)
  - applicabilité (« *compatible avec de nombreuses bases : MySQL, SQLite, Firebird, Oracle, etc.*)

Petit aperçu de PDO : création d'une instance

```php
<?php
// creation de l'instance PDO et connexion à la BD
$dsn = "mysql:host=$mysqlHost;" .
       "dbname=$mysqlDatabase;" .
       "charset=$charset";

// les options
$opt = array (
  PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
  PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
  PDO::ATTR_EMULATE_PREPARES => false );

$PDO = new PDO($dsn, $mysqlLogin, $mysqlPassword, $opt);
?>
```

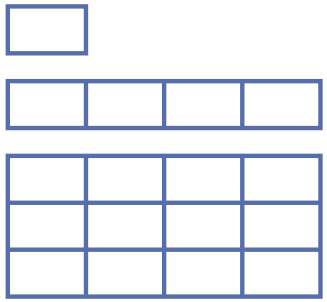
Petit aperçu de PDO : utilisation de l'instance

```php
<?php
// définition de la requête
$query = "SELECT * FROM $mysqlTable ".
         "WHERE field1=? AND field2 IN (?,?)";
$data = array ( 'f1', 3, 5 );

// envoi et exécution de la requête à la base
$statement = $PDO->prepare( $query ); // préparation
$exec = $statement->execute( $data ); // exécution

// récupération du résultat
$resultats = $statement->fetchAll ( PDO::FETCH_ASSOC );

// affichages
foreach ($resultats as $un_resultat)
  echo $un_resultat['field1'] ...;
?>
```

fetchColumn, fetch, fetchAll

- Récupérer des données de la BD

$\Longrightarrow$ {
- 1 valeur : `fetchColumn`
- 1 ligne : `fetch`
- plusieurs lignes : `fetchAll`



- Options pour `fetch` et `fetchAll` :
  - `PDO::FETCH_NUM` : indices = nombres
  - `PDO::FETCH_ASSOC` : indices = noms des champs dans la BD

`PDO::FETCH_NUM` :

```
array (
  0 => 'toto',
  1 => 'titi'
);
```

`PDO::FETCH_ASSOC` :

```
array (
  'nom' => 'toto',
  'prenom' => 'titi'
);
```

Quelques astuces

- **Documentation :**

  https ://www.php.net/manual/fr/funcref.php

- **Visualisation des erreurs :**

  En début de programme :

```
ini_set('display_errors', 'On');
error_reporting(E_ALL);
```

Javascript

Exécution de javascript

- Browsers contiennent un moteur Javascript :

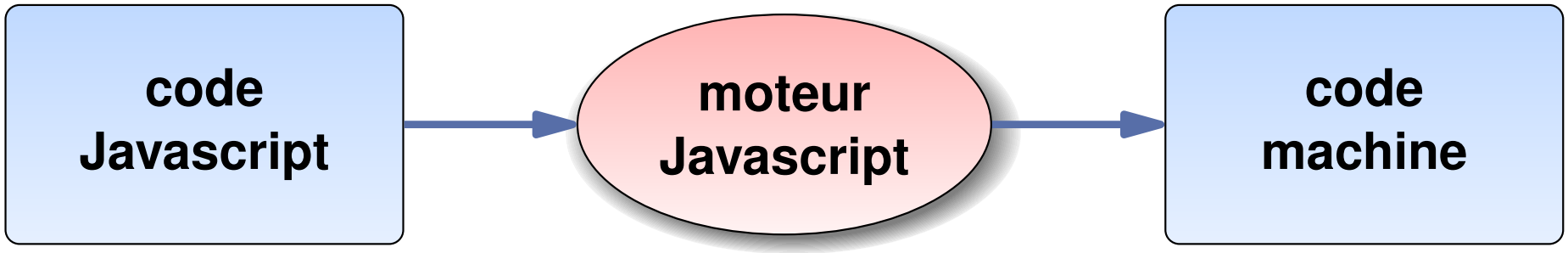

Chakra


Nitro


SpiderMonkey

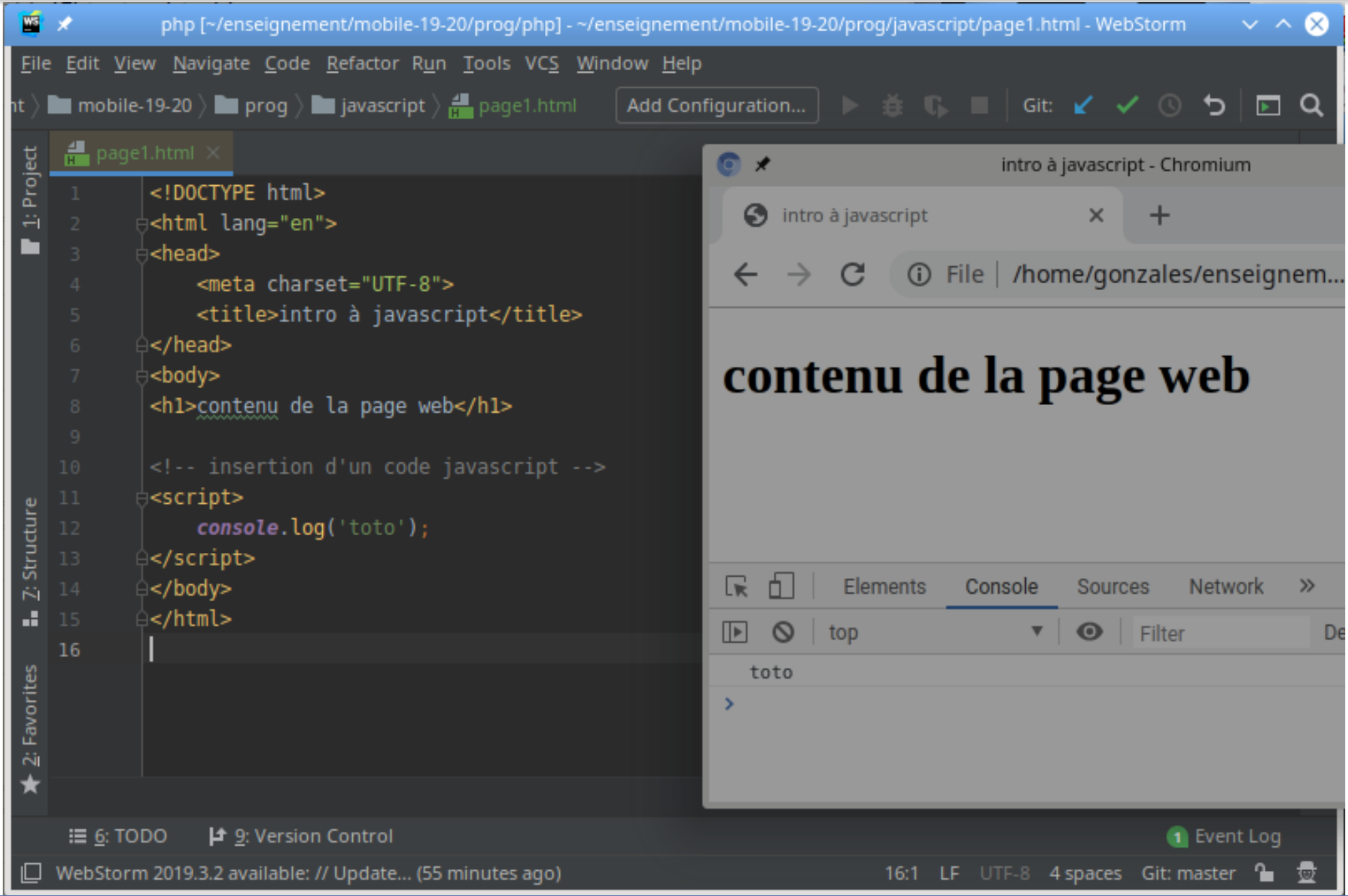

V8

- Vérifient les normes ECMAScript (ES6 = ES2015)
- Principe de fonctionnement :

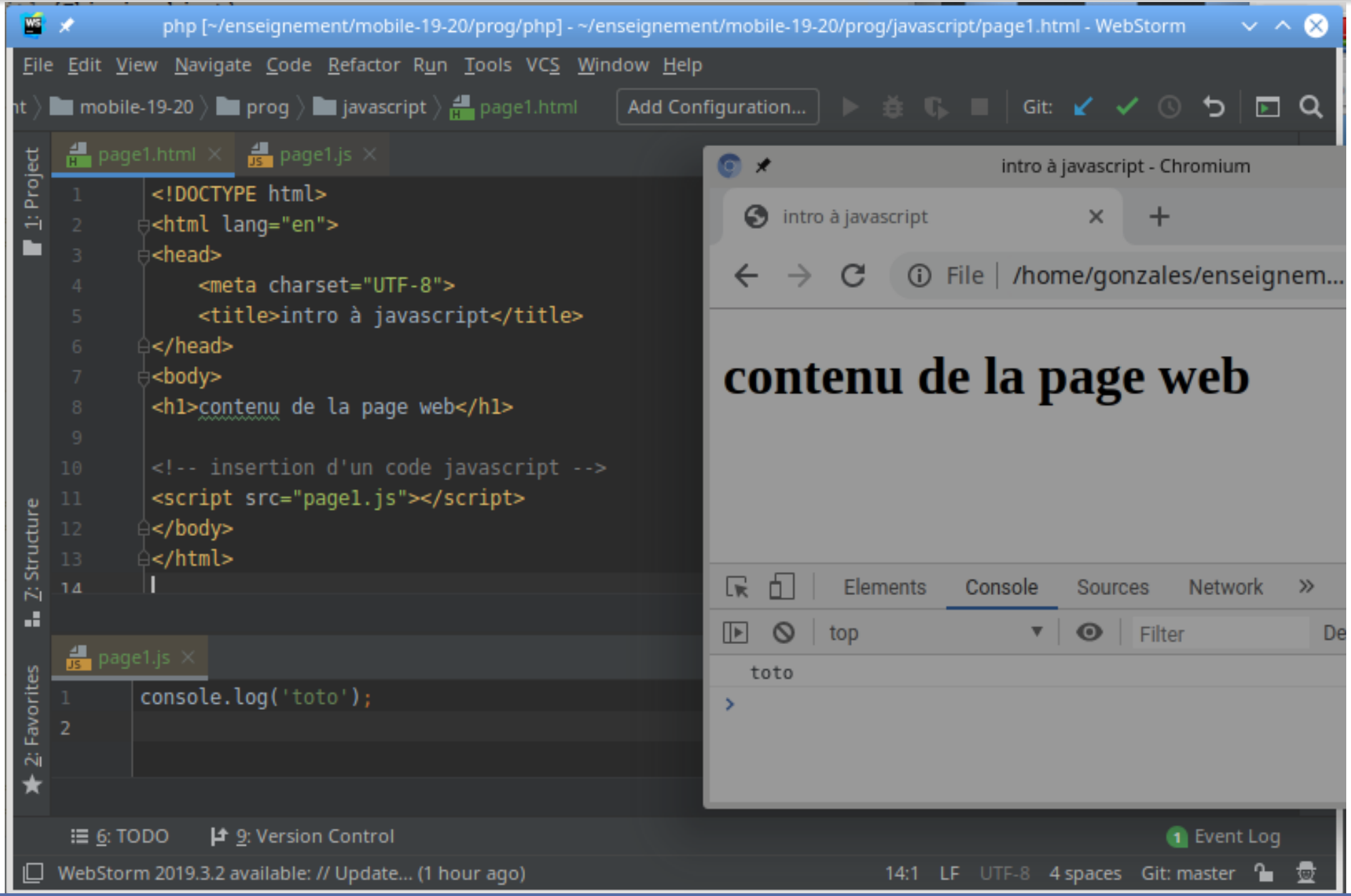


- Browser exécute ce code machine
- Depuis 2009 : compil/exec hors browser : 

HTML et javascript



Inclusion d'un fichier javascript



Création de variables en javascript : 4 manières

✔ `let nom_variable = valeur;`

`const nom_variable = valeur;`

portée de bloc si déclaration dans un bloc

portée de fichier/module sinon

~ `global.nom_variable = valeur;`

~ `window.nom_variable = valeur;`

création/modification d'une variable globale

✖ `nom_variable = valeur;`

modification de la valeur d'une variable si elle existe déjà

création d'une variable globale sinon

✖ `var nom_variable = valeur;`

portée de fonction si déclaration dans une fonction

variable globale si déclaration hors fonction

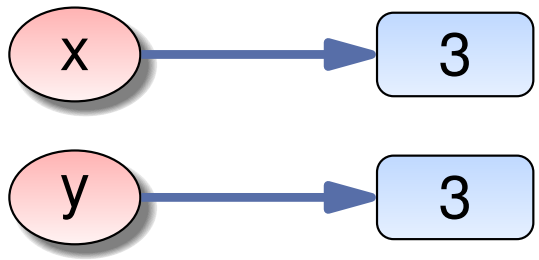
Types des variables (1/2)

Types primitifs :

- string : `let x = 'toto', y = 'to' + 'to';`
- number : `let x = 30, y = 4.5;`
- boolean : `let x = true;`
- undefined : `let x, y = undefined;`
- null : `let x = null;`

Types primitifs $\Longrightarrow$ copie par valeur :

```
let x = 3;
let y = x;
```



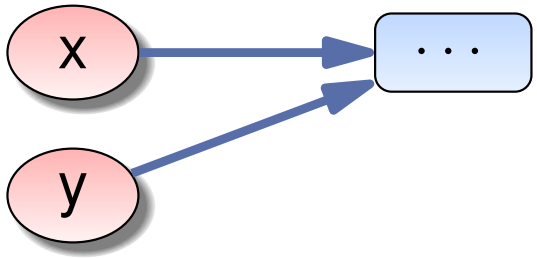
Types des variables (2/2)

Types référence :

- Object
- Array
- Function

Types référence $\Longrightarrow$ copie par référence :

```
let x = ...;
let y = x;
```



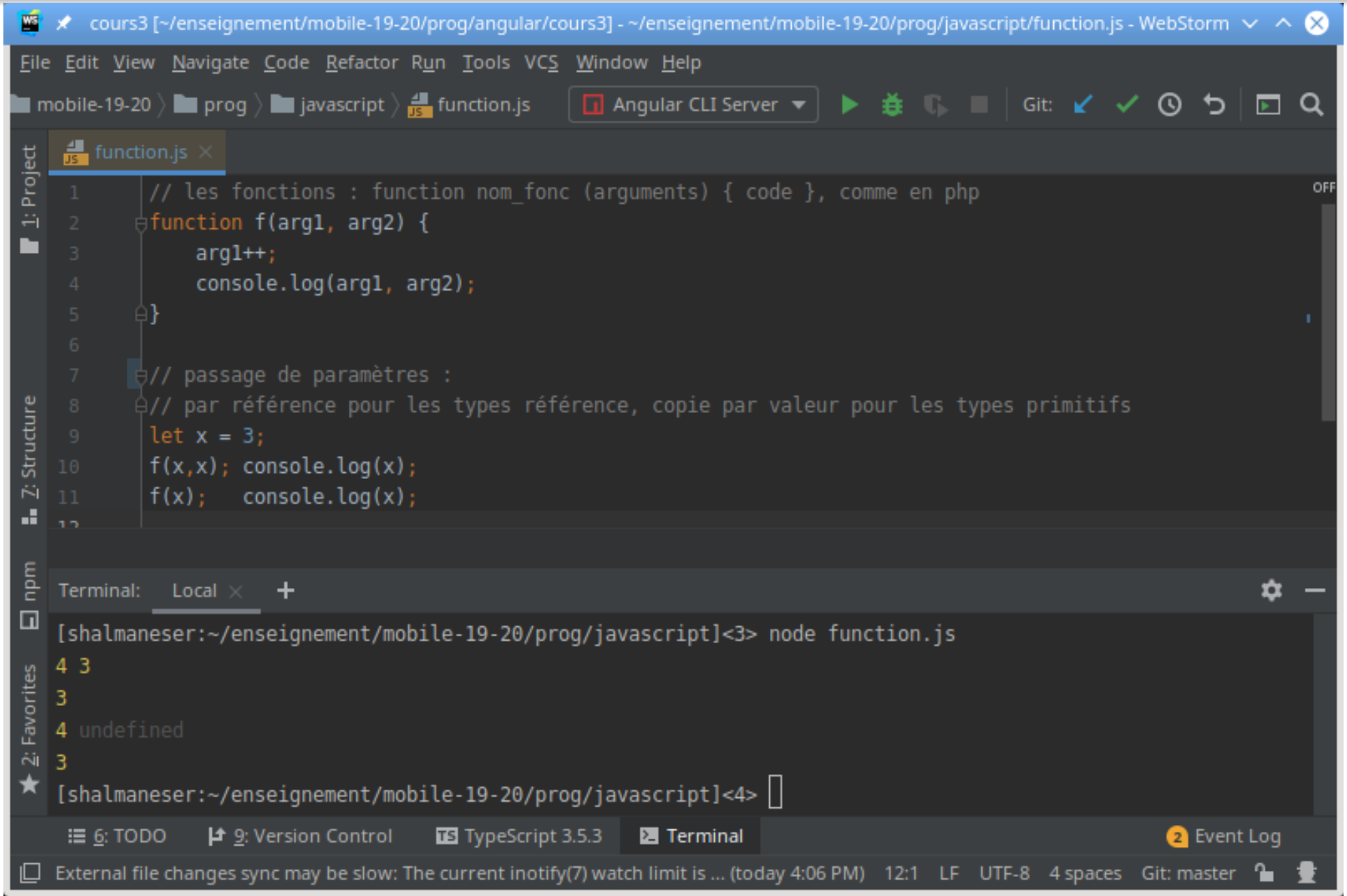
Noms des variables

- **Noms autorisés :** mêmes règles qu'en Java ou C
- **Convention :** notation Camel

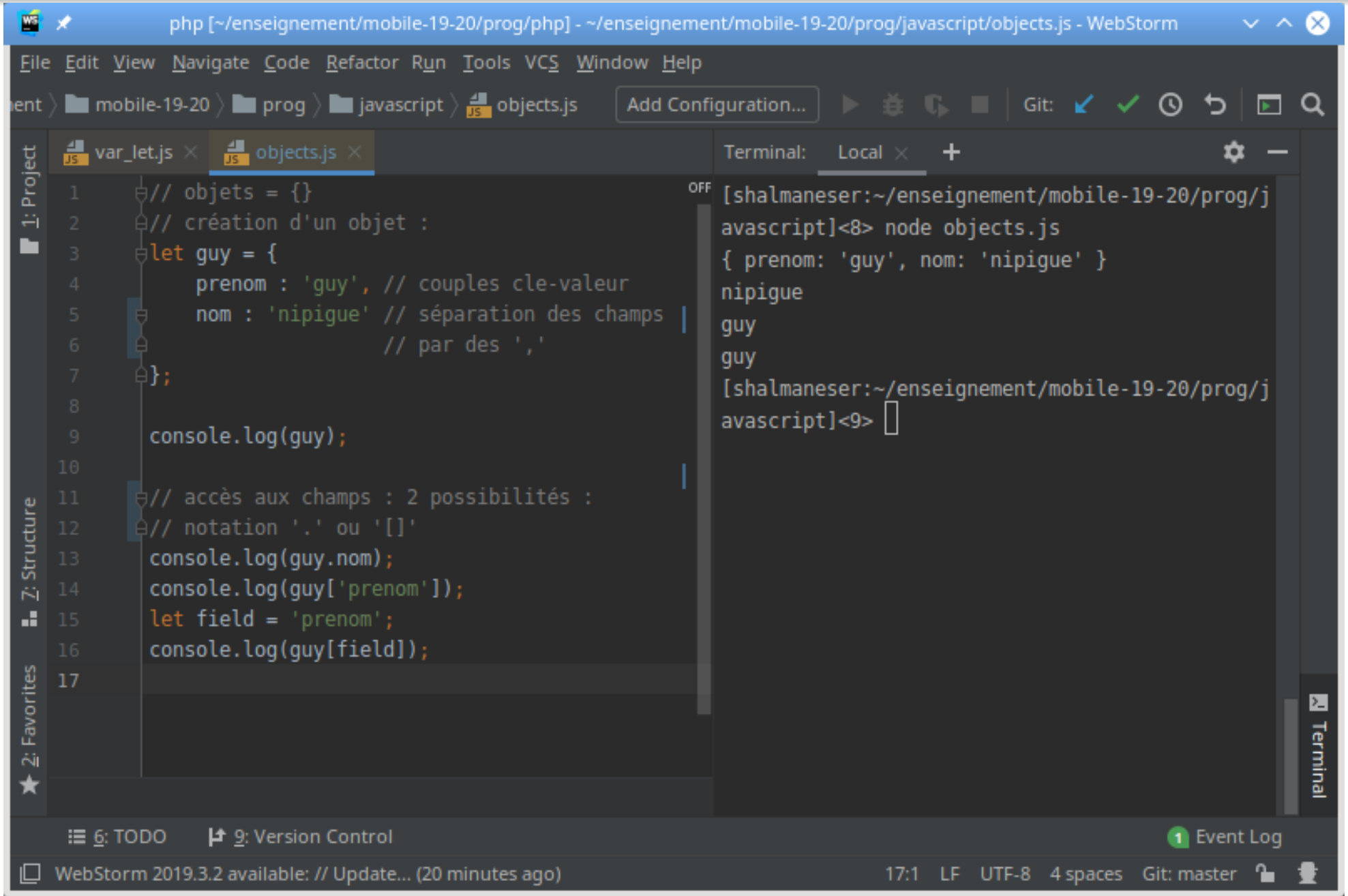
  Exemple : `firstName`
- **Attention :** Javascript sensible à la casse :

  Exemple : `firstName` $\neq$ `FirstName`

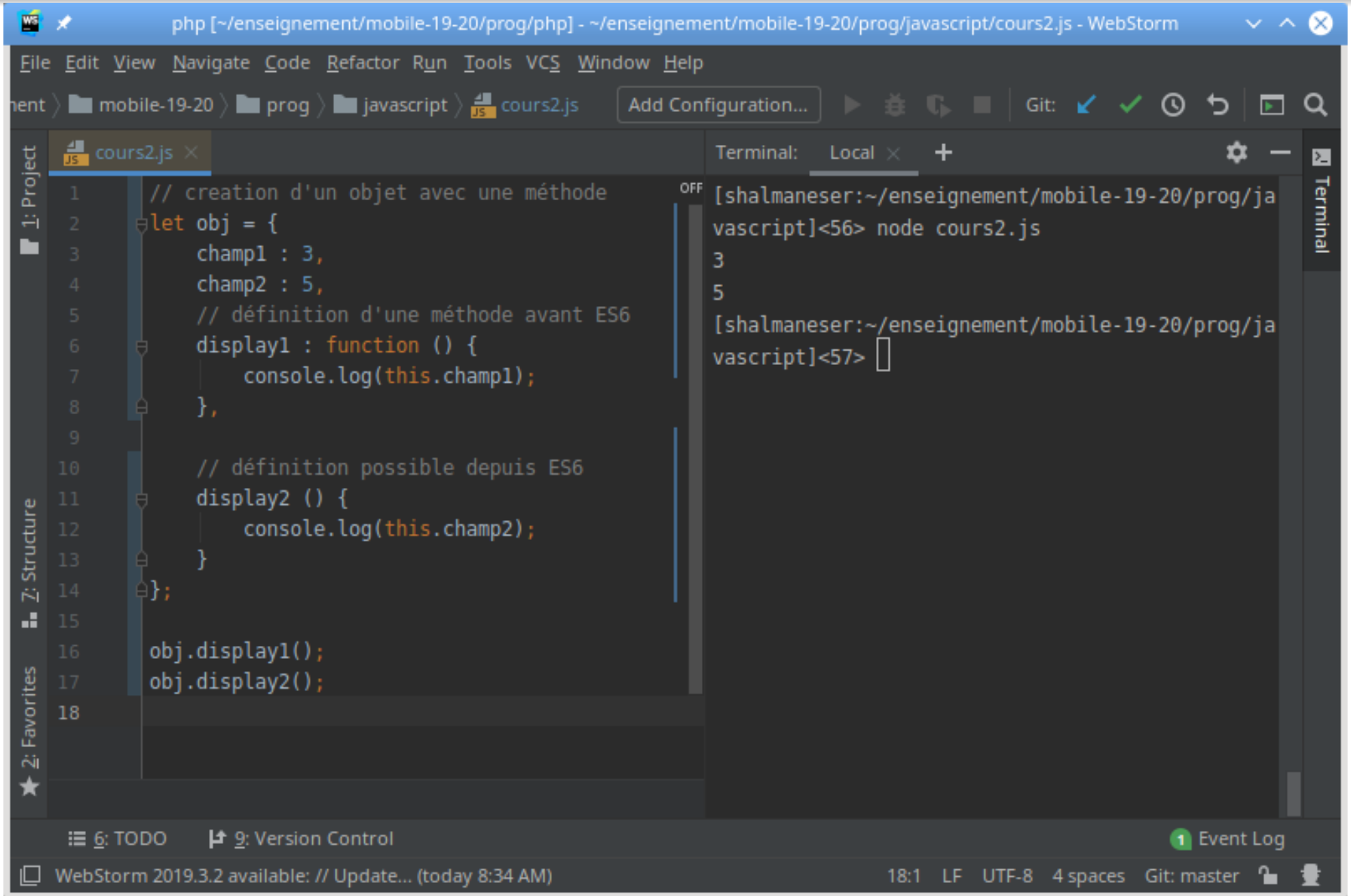
Les fonctions



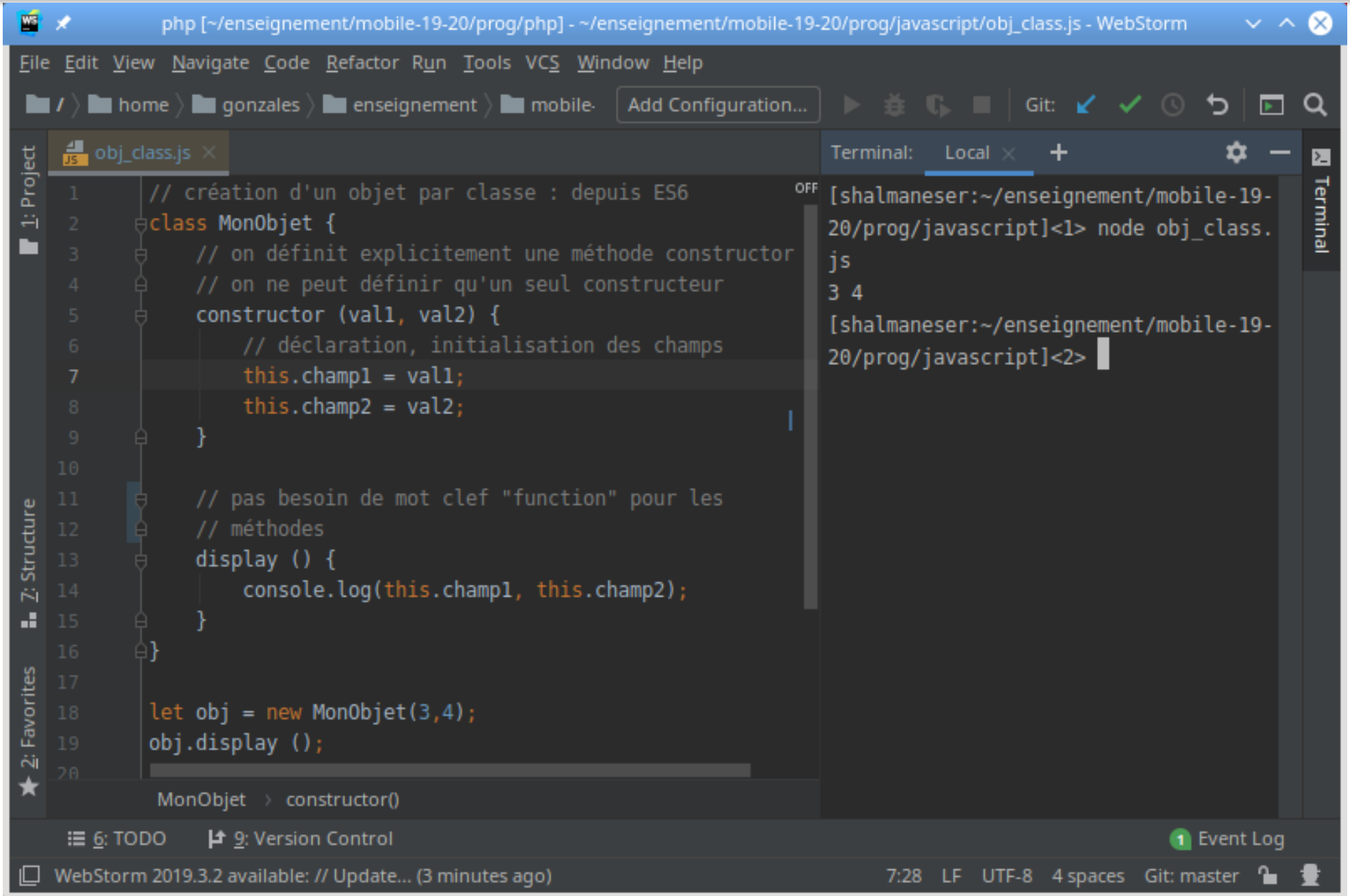
Les objets (1/3)



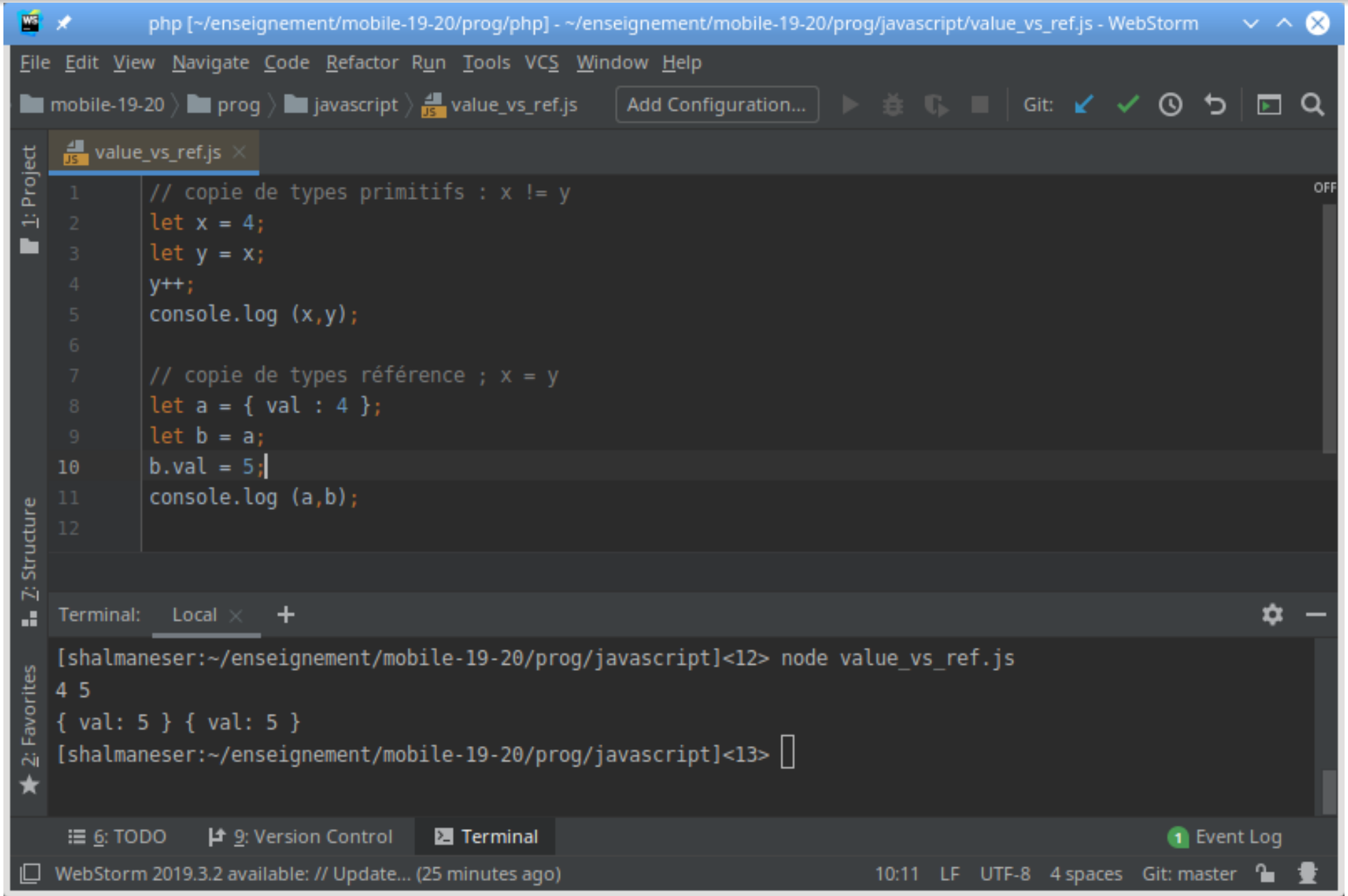
Les objets (2/3)



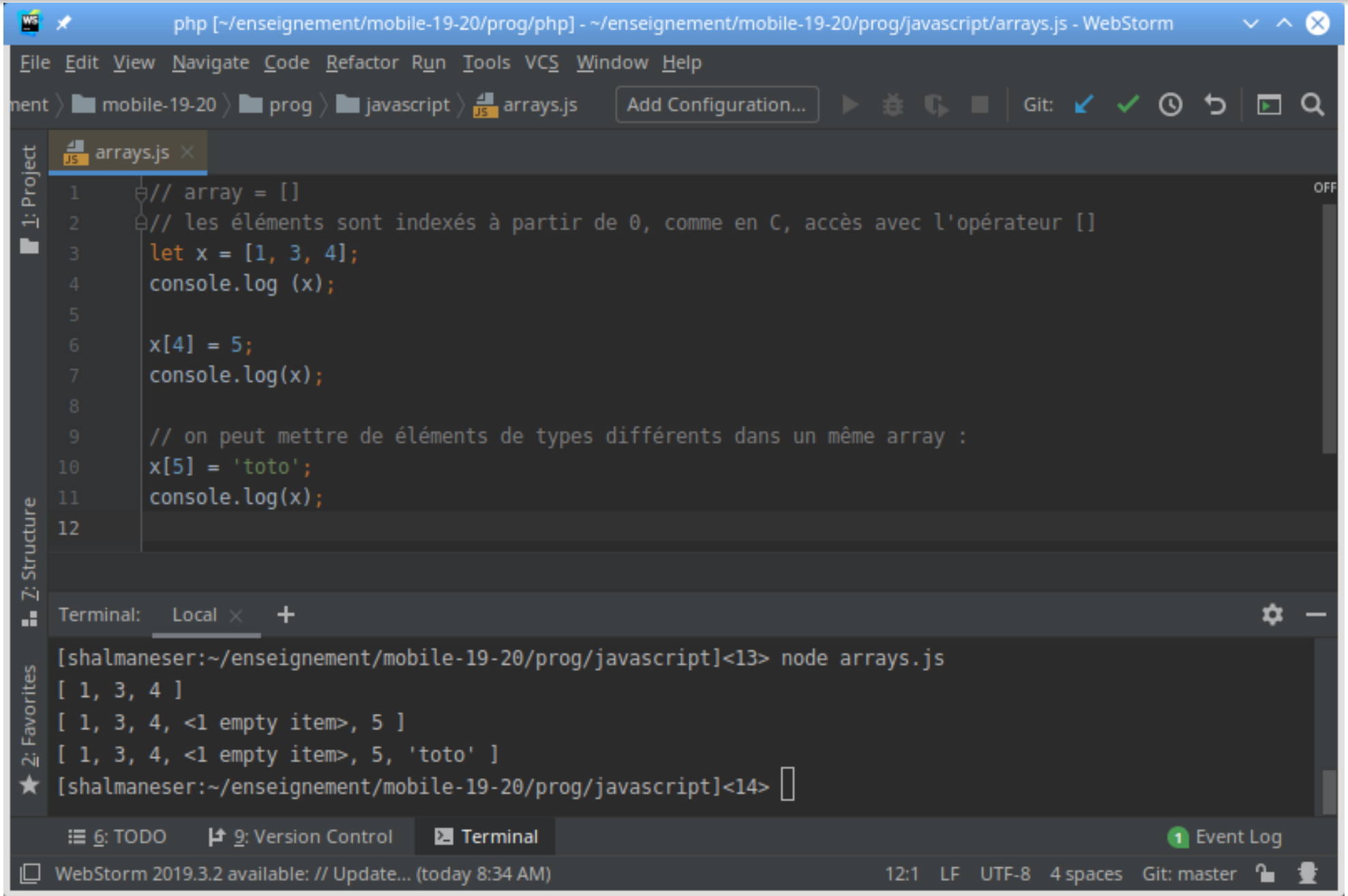
Les objets (3/3)



Comparaison types primitifs / référence

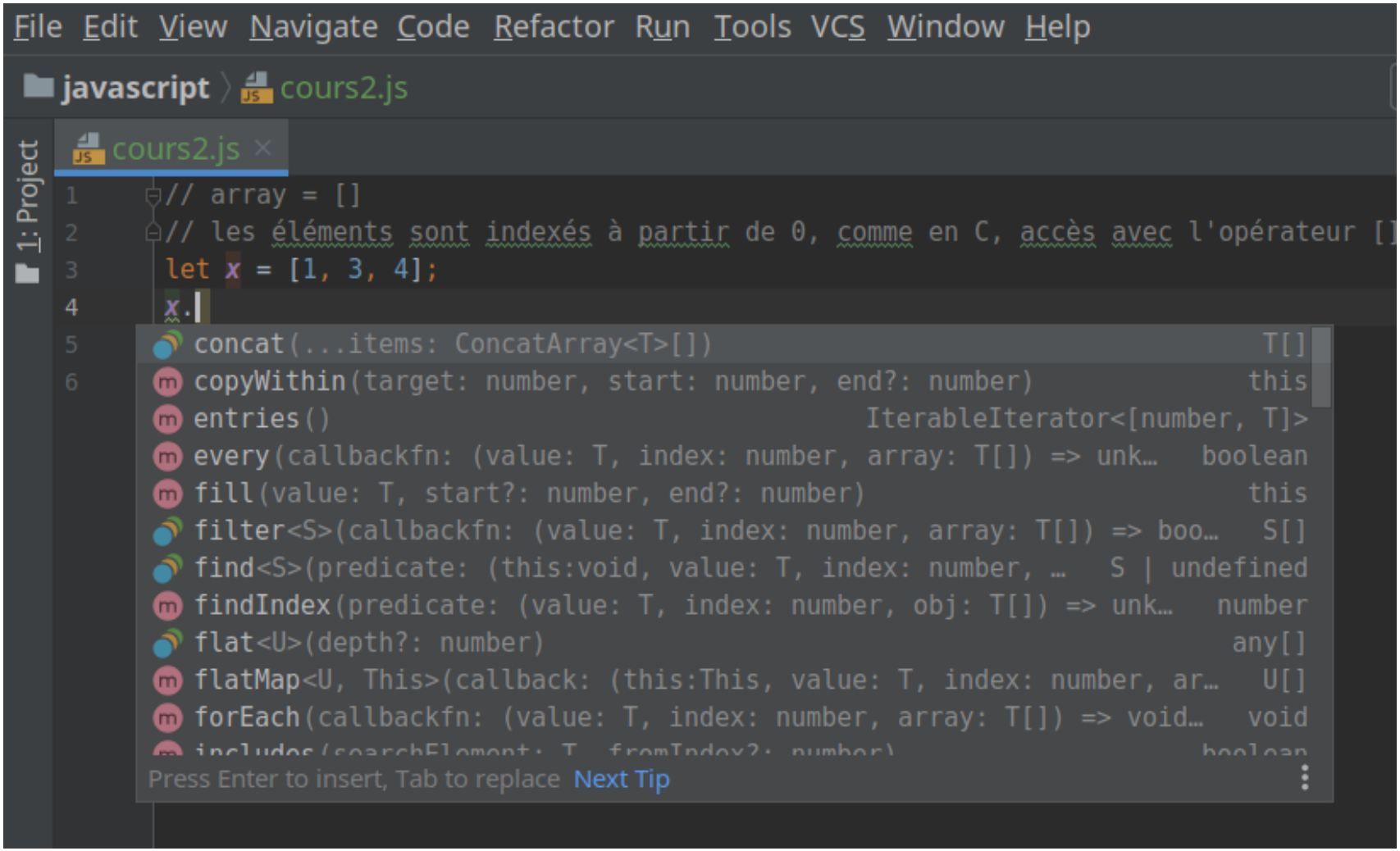


Les tableaux (1/2)



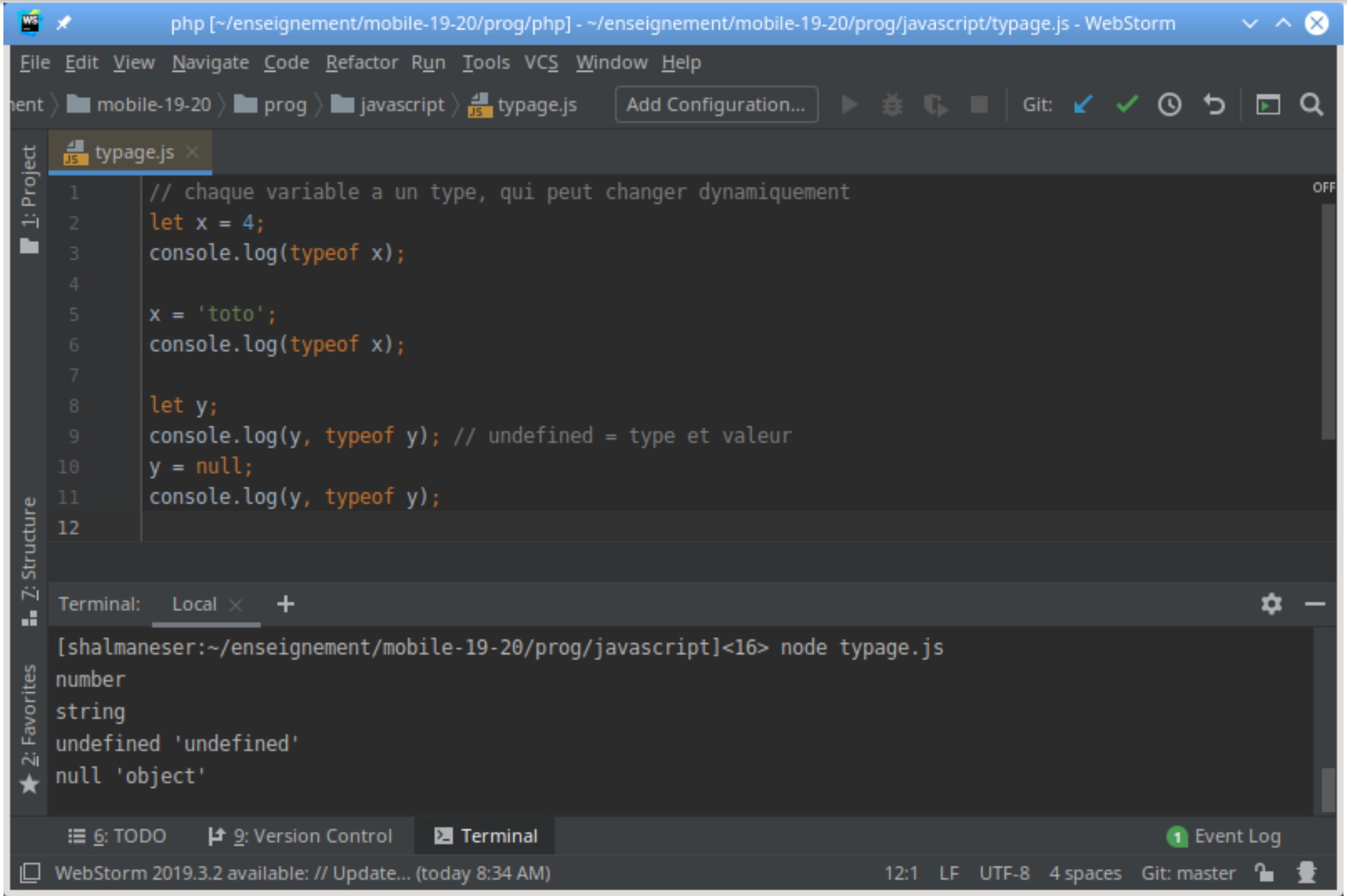
Les tableaux (2/2)

Les tableaux sont des objets :

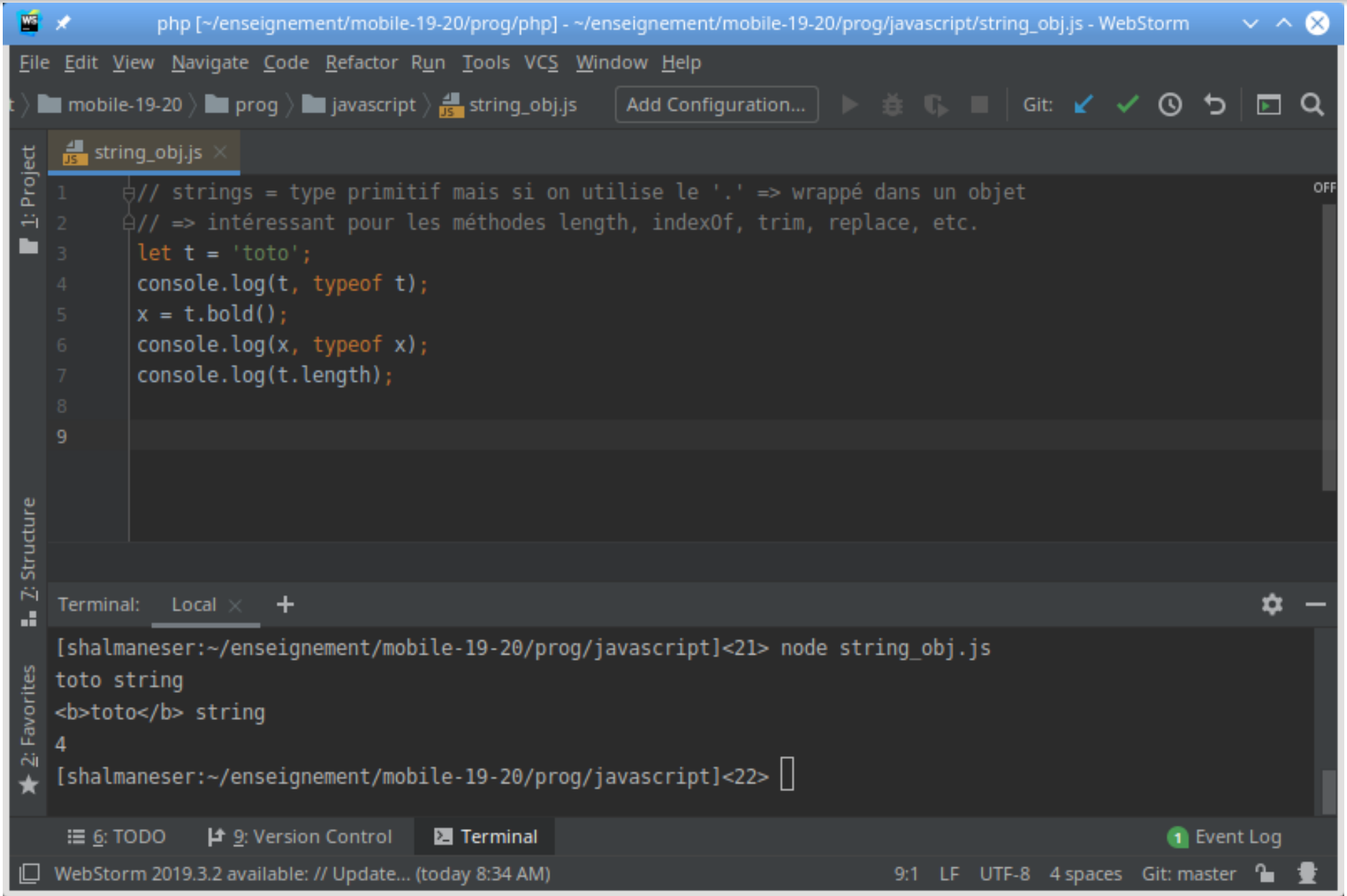


$\Longrightarrow$ contiennent des méthodes (length, filter, forEach, *etc.*)

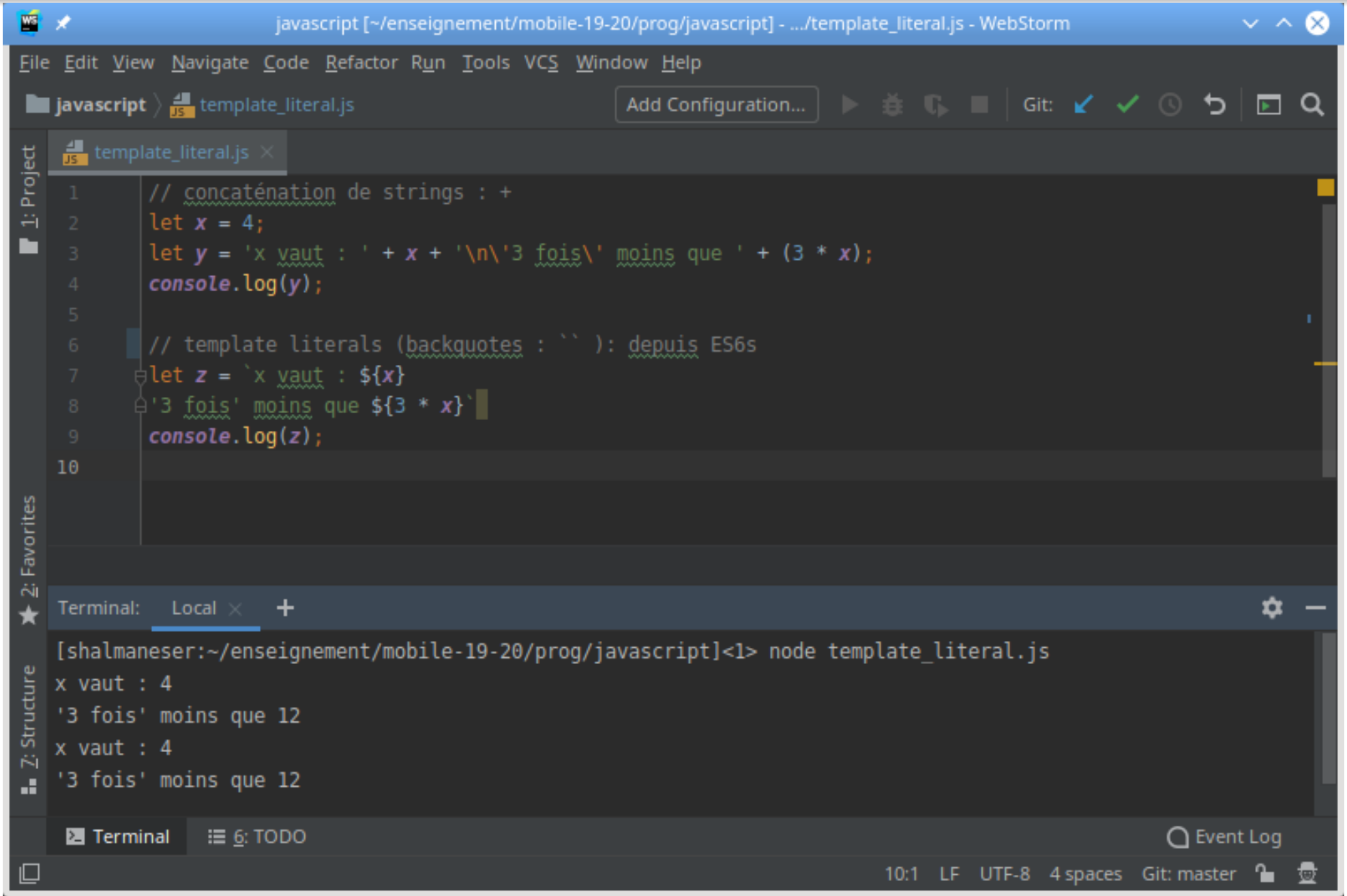
Typage dynamique



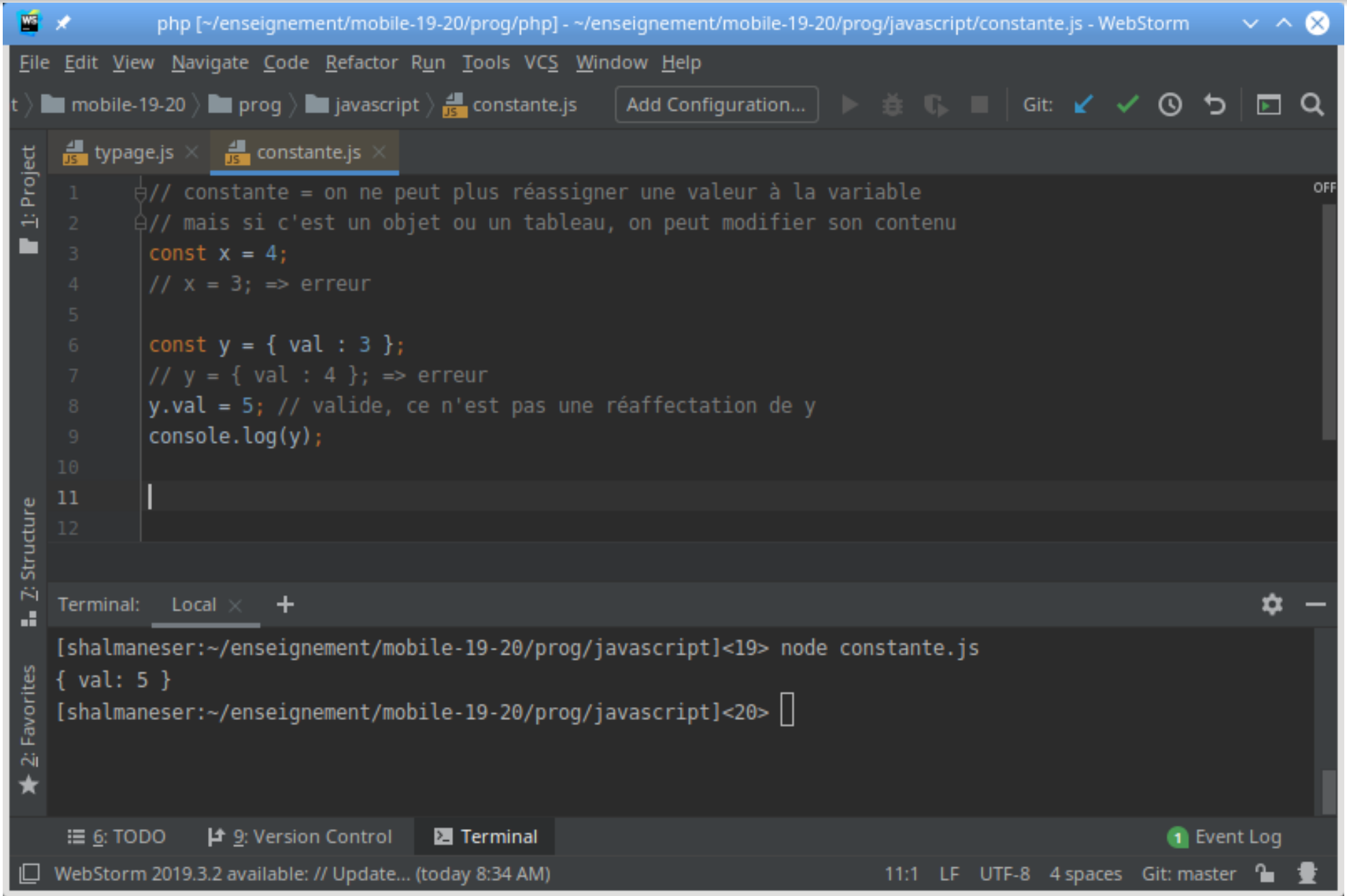
Retour sur les string et les objets



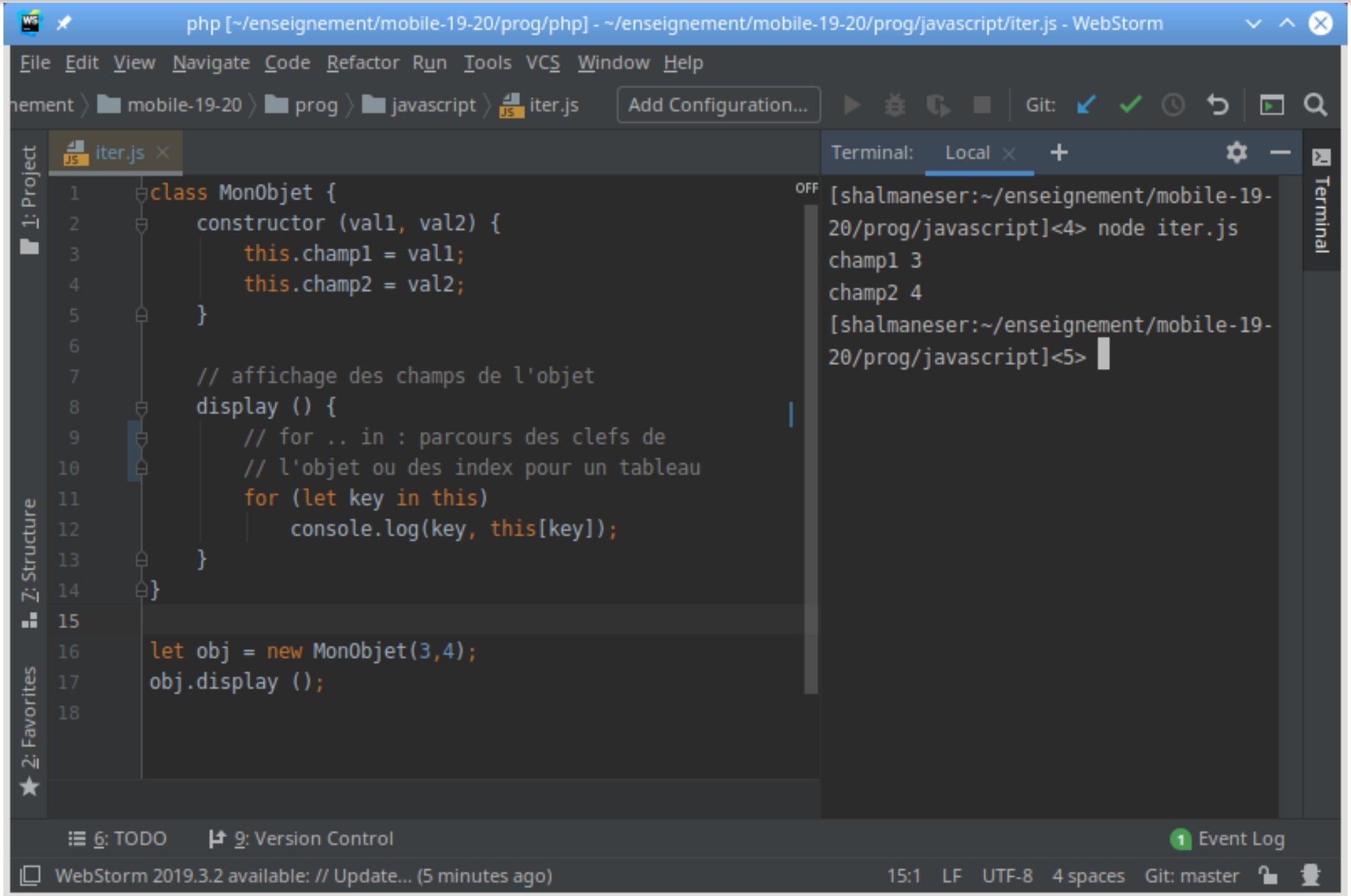
Template literals



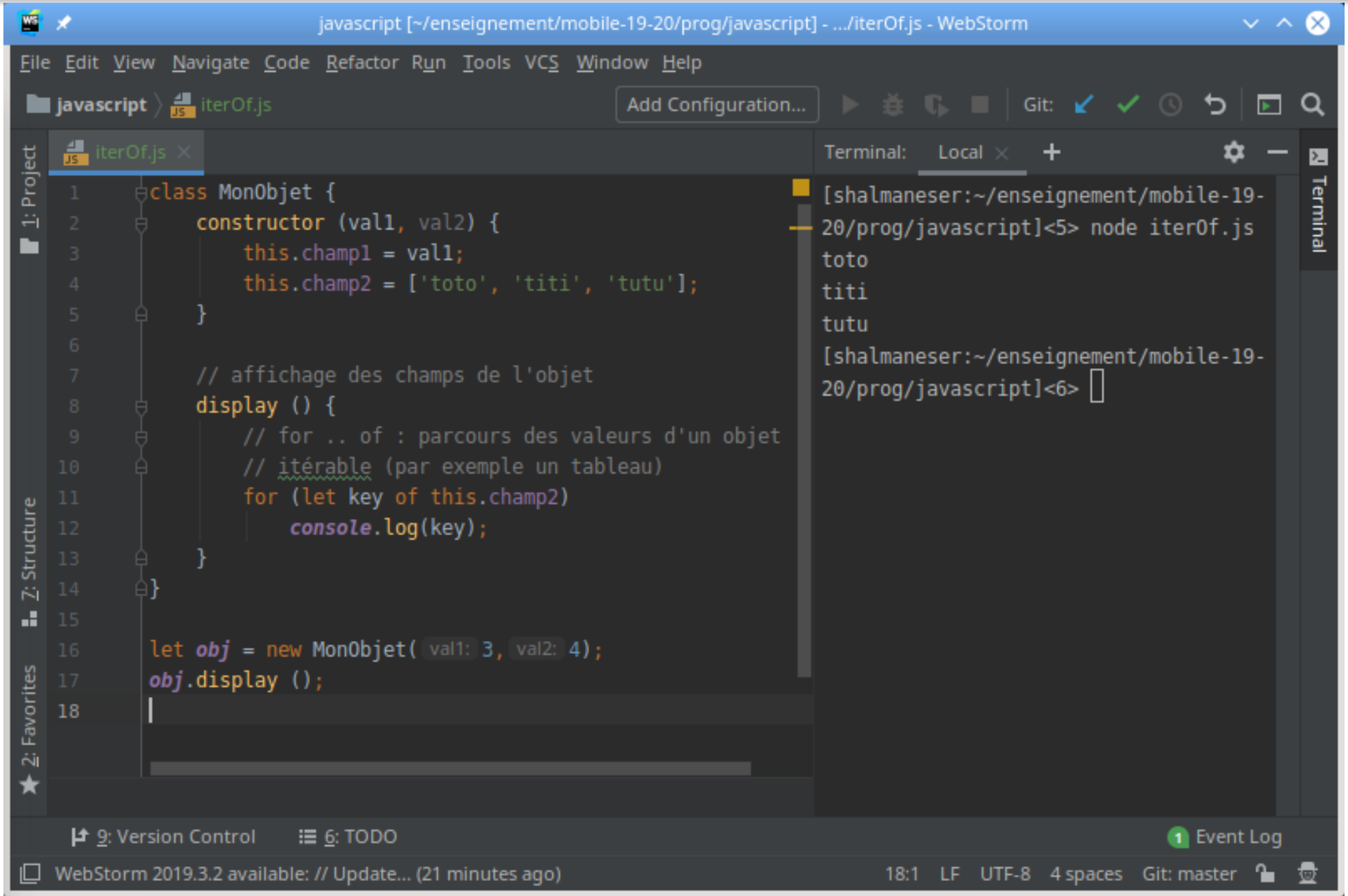
Les constantes



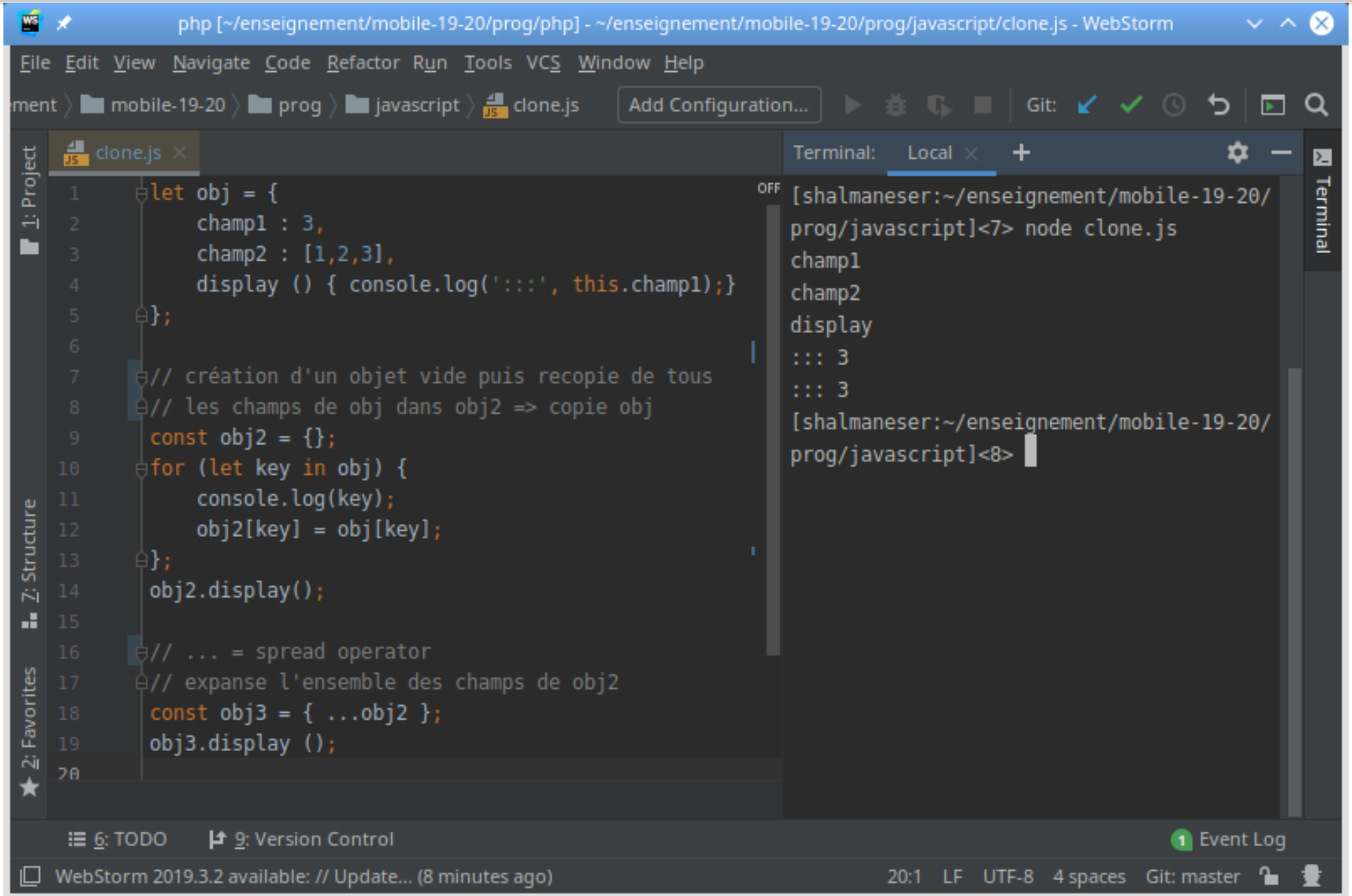
Itérer les champs d'un objet (1/2)



Itérer les champs d'un objet (2/2)

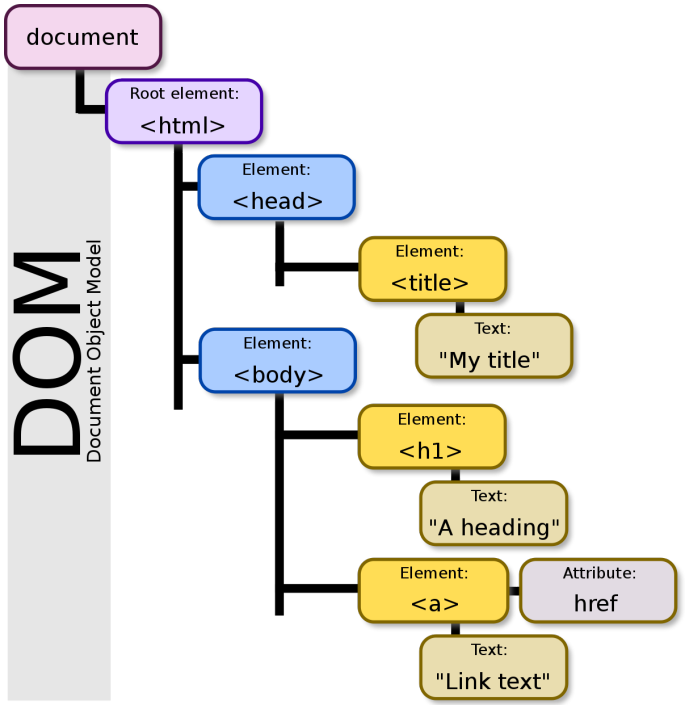


Copier (cloner) un objet



Javascript et le DOM

DOM : Document Object Model



[image wikipedia]

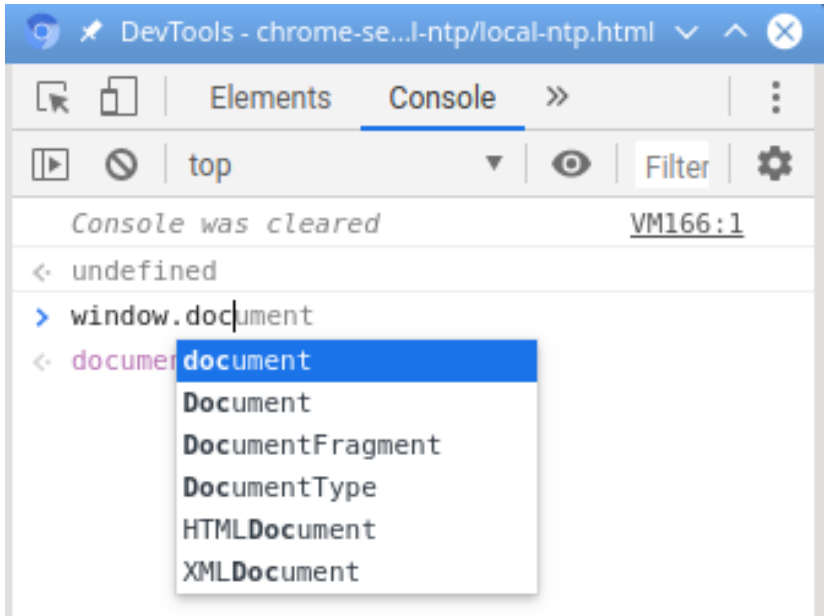
- Représente la page web affichée
- Structure d'arbre — racine : `document`

Javascript : window vs document

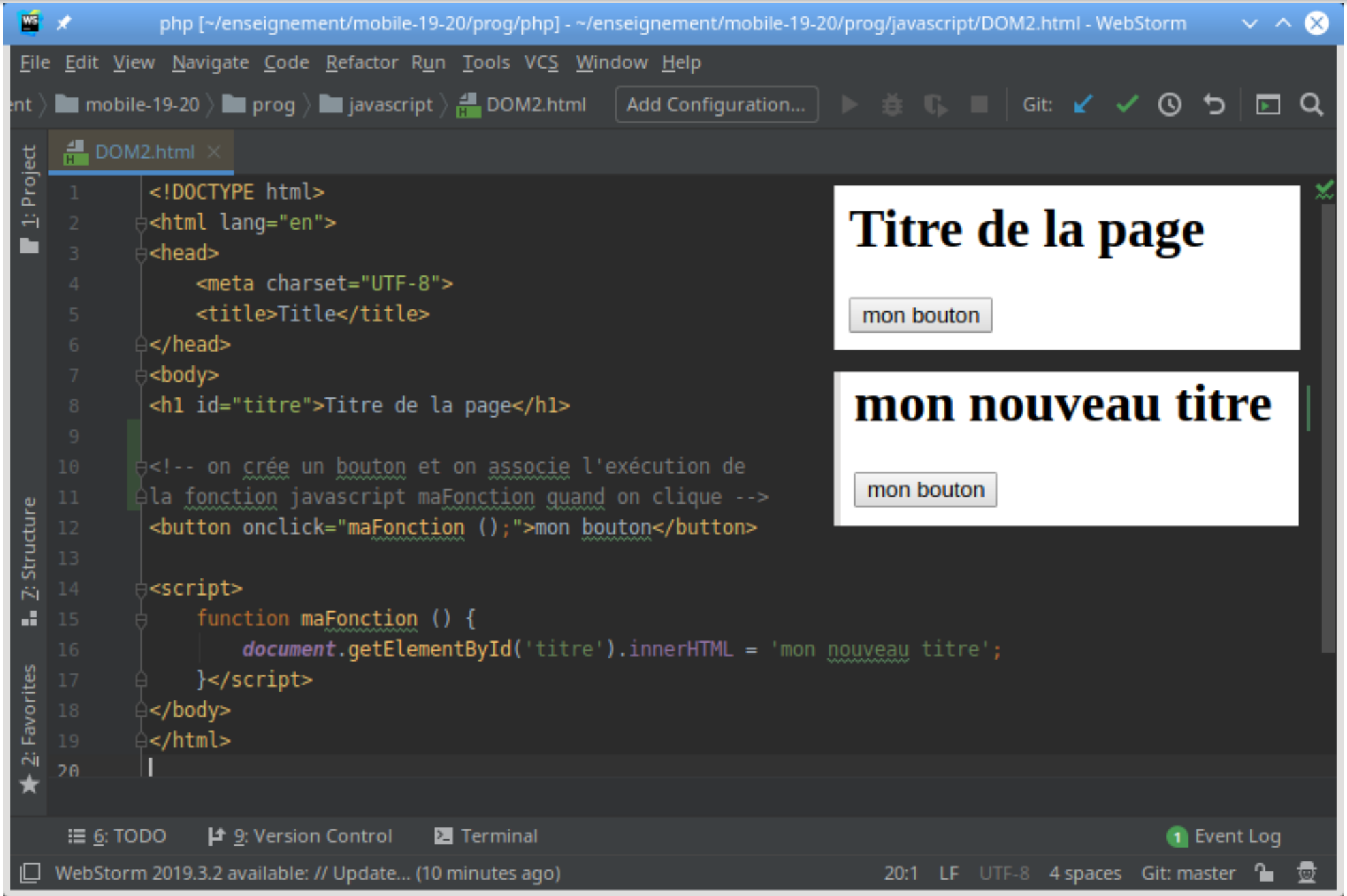
Pour Javascript :

- Dans un browser, `window` = objet global
- `document` = `window.document`

  $\Longrightarrow$ `document` est une propriété de `window`
- `document` = l'objet « visible » du browser



Manipuler le DOM



Rajouter des éléments au DOM