

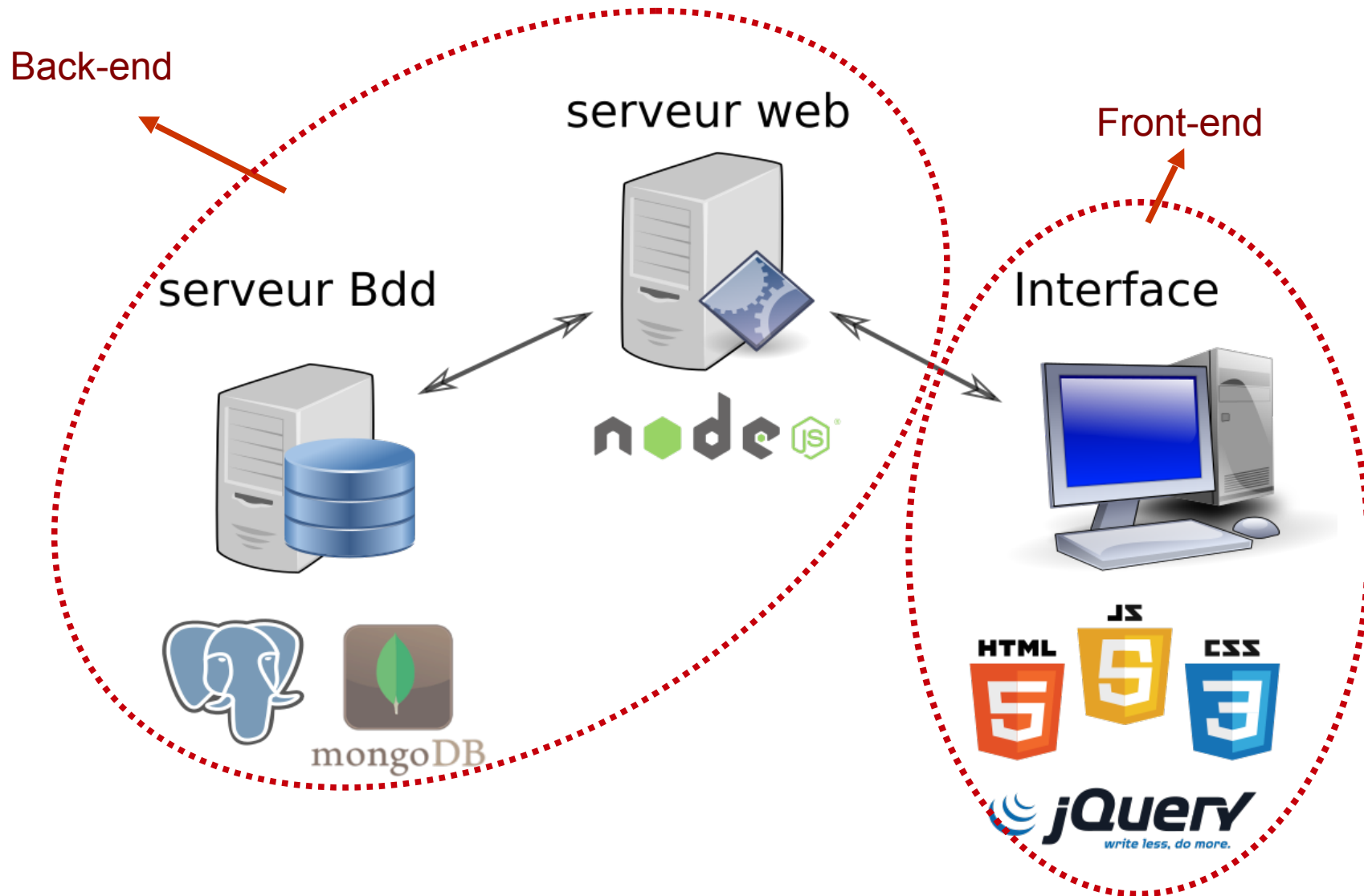
Découverte de différents outils JavaScript « front-end »

François Agneray



Application web

Architecture





JavaScript « front-end »

Permet de développer des applications Internet riches (RIA)

Types de fonctionnalités des RIA :

- Glisser-déposer
- Barres d'outils et rafraichissement automatique
- Calculs

Quelques exemples de RIA :

- Gmail : courriel
- Deezer : site d'écoute musicale
- Airbnb : location et réservation de logements

Avantages :

- Aucune installation nécessaire
- Nécessite simplement une connexion Internet et un navigateur récent





Exemple simple

todos

▼ *What needs to be done?*

Todo 1

Todo 2

Todo 3

3 items left Active Completed

Double-click to edit a todo

Created by petehunt

Part of TodoMVC

Une bibliothèque de fonctions JavaScript pour le parcours et la modification du DOM des pages web

Jquery contient les fonctionnalités suivantes :

- Parcours et modification du DOM
- Événements
- Ajax
- Utilisation et développement de Plugins
- Manipulation des CSS
- Effets visuels et animations

Informations

- Dernière version : 1.1.3 et 2.1.4
- Première release : 26 aout 2006
- Dernière release : 28 avril 2015
- Licence : MIT
- Site web : <https://jquery.com>

Afficher une « todo list » en jQuery

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Todo list</title>
    <link rel="stylesheet" href="css/app.css" />
  </head>
  <body>

    <form>
      <label>Enter a Todo and hit enter!</label>
      <input type="text" /> <!-- L'utilisateur saisi un todo + entrée -->
    </form>

    <ul></ul> <!-- Au démarrage la liste est vide -->

    <script src="bower_components/jquery/dist/jquery.min.js"></script>
    <script src="js/app.js"></script>
  </body>
</html>
```

Afficher une « todo list » en jQuery

```
$(document).ready(function () {  
  
  $('form').submit(function () {  
    if ($('#input').val() !== "") {  
      var input_value = $('#input').val();  
      // On modifie le DOM à chaque nouveau todo  
      $('#ul').append('<li>' + input_value + '<a href="">X</a></li>');  
    }  
    $('#input').val("");  
    return false;  
  });  
  
  $(document).on('click', 'a', function (e) {  
    e.preventDefault();  
    $(this).parent().remove();  
  });  
  
});
```

Un framework JavaScript pour l'écriture d'une application complète

Les principales caractéristiques d'AngularJS sont :

- Application web mono-page
- Propose une organisation pour le code
- Extension du langage HTML
- Patron de conception MVC
- Data-Binding
- Intègre une version allégé de jQuery (jQLite)

Informations

- Dernière version : 1.4.8
- Première release : 2009
- Dernière release : 19 novembre 2015
- Licence : MIT
- Site web : <https://angularjs.org>

Afficher la même todolist avec AngularJS

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Todo list</title>
    <link rel="stylesheet" href="css/app.css" />
  </head>
  <body ng-app="todo" ng-controller="TodoController">

    <form ng-submit="submit()">
      <label>Enter a Todo and hit enter!</label>
      <input type="text" ng-model="todo" /> <!-- Todo + entrée -->
    </form>

    <ul ng-repeat="todo in todos">
      <li>{{todo}}<a ng-click="remove(todo)">X</a></li>
    </ul>

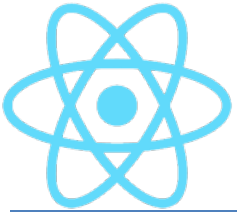
    <script src="bower_components/angular/angular.min.js"></script>
    <script src="js/app.js"></script>
  </body>
</html>
```

Afficher la même todolist avec AngularJS

```
angular.module('todo', []);
angular.module('todo').controller('TodoController', function ($scope) {
  // Au lancement la liste est vide
  $scope.todos = [];

  // Ajout d'un TODO
  $scope.submit = function () {
    if($scope.todo) {
      $scope.todos.push($scope.todo);
      $scope.todo = "";
    }
  };

  // Suppression d'un TODO
  $scope.remove = function (todo) {
    var index = $scope.todos.indexOf(todo);
    if(index > -1) {
      $scope.todos.splice(index, 1);
    }
  }
});
```



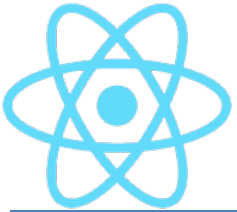
Un framework pour la gestion de la partie Vue d'une application web mono-page

Les principales caractéristiques de ReactJS sont :

- Application web mono-page
- Permet de créer des composants web réutilisables
- Ne gère que l'interface de l'application (IHM)
- Peut être utilisé avec un autre framework (AngularJS, Backbone...)
- JSX un langage pour faciliter l'écriture de la vue

Informations

- Dernière version : 0.14
- Première release : 2013
- Dernière release : 7 octobre 2015
- Licence : BSD
- Site web : <http://reactjs.com>

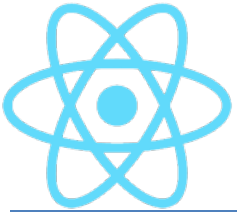


Afficher la todolist avec ReactJS

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Todo list</title>
    <link rel="stylesheet" href="css/app.css" />
  </head>
  <body>

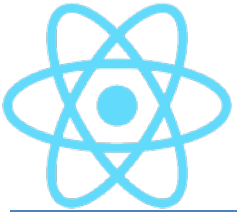
    <div id="app"></div>

    <script src="bower_components/react/ract.min.js"></script>
    <script src="bower_components/react/react-dom.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/babel-core/5.8.23/browser.min.js"></script>
    <script type="text/babel" src="js/app.js"></script>
  </body>
</html>
```



Afficher la todolist avec ReactJS

```
var TodoList = React.createClass({
  render: function () {
    var createItem = function(item) {
      return <li key={item.id}>{item.text}</li>;
    };
    return <ul>{this.props.items.map(createItem)}</ul>;
  }
});
```



```
var TodoApp = React.createClass({
  getInitialState: function () {
    return {items: [], text: ""};
  },
  onChange: function (e) {
    this.setState({text: e.target.value});
  },
  handleSubmit: function (e) {
    e.preventDefault();
    var nextItems = this.state.items.concat([{text: this.state.text, id: Date.now()}]);
    this.setState({items: nextItems, text: ""});
  },
  render: function () {
    return (
      <div>
        <h3>TODO</h3>
        <form onSubmit={this.handleSubmit}>
          <input onChange={this.onChange} value={this.state.text} />
        </form>
        <TodoList items={this.state.items} />
      </div>
    )
  }
});
ReactDOM.render(<TodoApp />, app);
```

Comparatif

Les avantages et inconvénients de chaque framework



Permet de manipuler facilement le DOM

+

- Beaucoup de ressources disponibles
- Prise en main facile

-

- Ne permet pas de développer une application complète
- Pas de « data-binding »



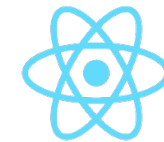
Développement d'une application « front-end »

+

- Data-binding
- Application MVC complète
- Intégration de jQuery

-

- Temps d'apprentissage



React

Programmation orientée composant

+

- Découplage de la partie Vue

-

- Très peu de documentation
- Temps d'apprentissage

JavaScript « Front-end »

Framework JavaScript MVC



Bibliothèque de fonctionnalités

