ReactJS - Maîtriser le framework

Ihab ABADI

Programme Partie 1

- Rappels sur notion avancée Javascript
- Rappels HTML5, CSS3, Le DOM
- Le Pattern MVC
- ReactJs definition
- Virtual DOM
- Installation outils de développement
- Javascript ES6 (ES2015)

Programme Partie 2

- Composant React JS
- Création d'un composant
- JSX
- Propriétés d'un composant
- Etat d'un composant
- Cycle de vie d'un composant

Programme Partie 3

- Gestion des évènements
- Communication Inter-Composant
- Auto-Binding react js
- Routing react js
- Gestion Formulaire

Programme Partie 4

- Isomorphisme react js
- React Native

Rappels sur notion avancée Javascript

-Le JavaScript, un langage orienté objet, tout est avant tout un objet

- chaîne de caractère, nombre, fonction, tableau…

- On peut créer un objet :
  1 - En utilisant ‘new Object()’
  2- D’une manière littérale
  3 - A partir d’un constructeur

```
var chaine = "Bonjour tout le monde";
var personne = {
    nom : "abadi",
    prenom : "ihab",
    age : 30,
    afficher : function() {
        return 'Nom : '+ this.nom+ ' Prénom : '+ this.prenom
    }
}
```

Rappels sur notion avancée Javascript

La portée des variables en Javascript :

- Variable Locale
- Variable globale

Rappels sur notion avancée Javascript

Fonction anonyme en Javascript :

- Une fonction anonyme en javascript est une fonction qui ne possède pas de nom.
- Plusieurs façons d'invoquer une fonction anonyme
- Exemple d'utilisation des fonctions anonymes

Rappels sur notion avancée Javascript

Les closures en Javascript :

- Une closure (de « close », « fermé » en anglais) est une fonction qui va récupérer et pouvoir utiliser des variables d'environnement dans laquelle elle a été créée.
- une closure va nous permettre d'isoler ou d'enfermer une partie de code et de pouvoir utiliser et récupérer des variables qui ne sont accessibles normalement que depuis l'intérieur de ce code
- Exemples d'utilisation de closure.

Rappels sur notion avancée Javascript

Notion de prototype en Javascript :

- Javascript se base sur un modèle prototypal et non pas classique (au sens « modèle à base de classes »). Le prototype d'un objet est utilisé pour fournir de façon dynamique des propriétés aux objets qui héritent du prototype
- Exemple d’utilisation de prototype

Rappels HTML5, CSS3, Le DOM

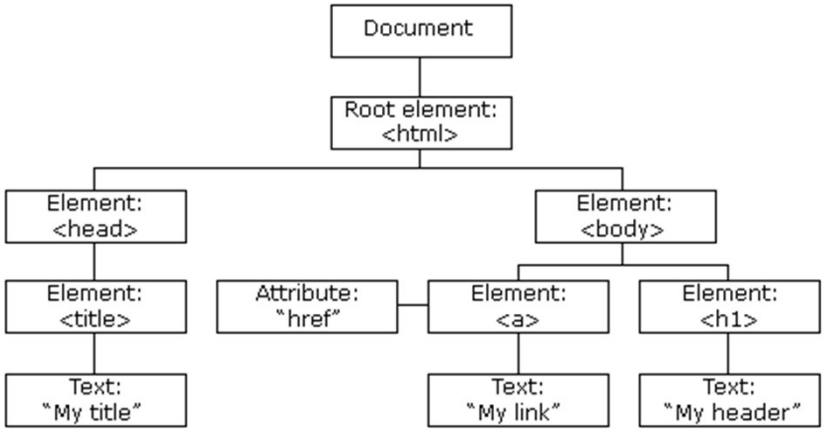
- Rappel HTML5
- Rappel CSS3

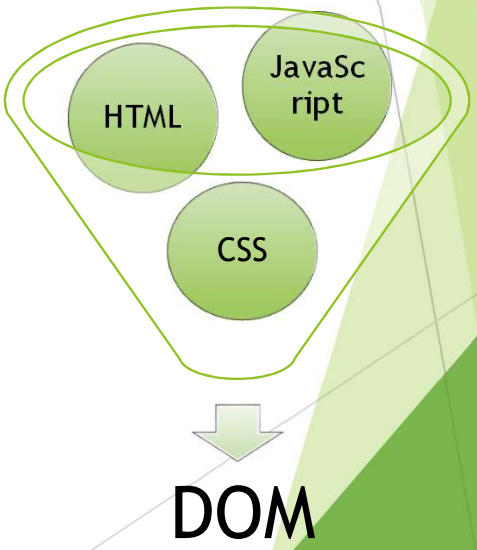
Rappels HTML5, CSS3, Le DOM

Définition du DOM :

- Document Object Model est le principe de transformer chaque éléments de notre application en un objet, pour pouvoir manipuler, son aspect, son comportement, ses évènements

- Cl

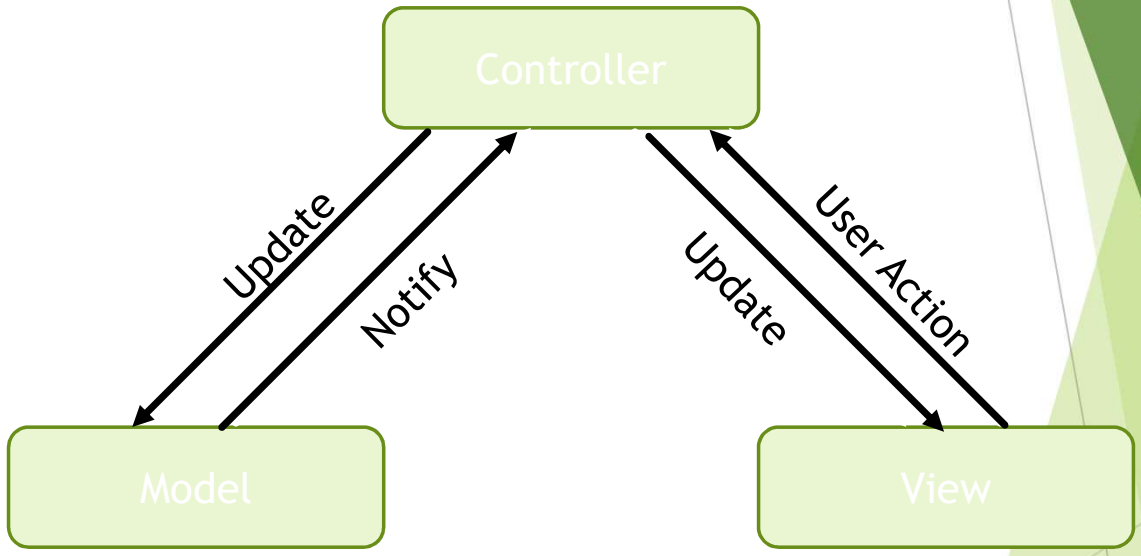




Rappels HTML5, CSS3, Le DOM

Le Pattern MVC:

- Controller
- Model
- View

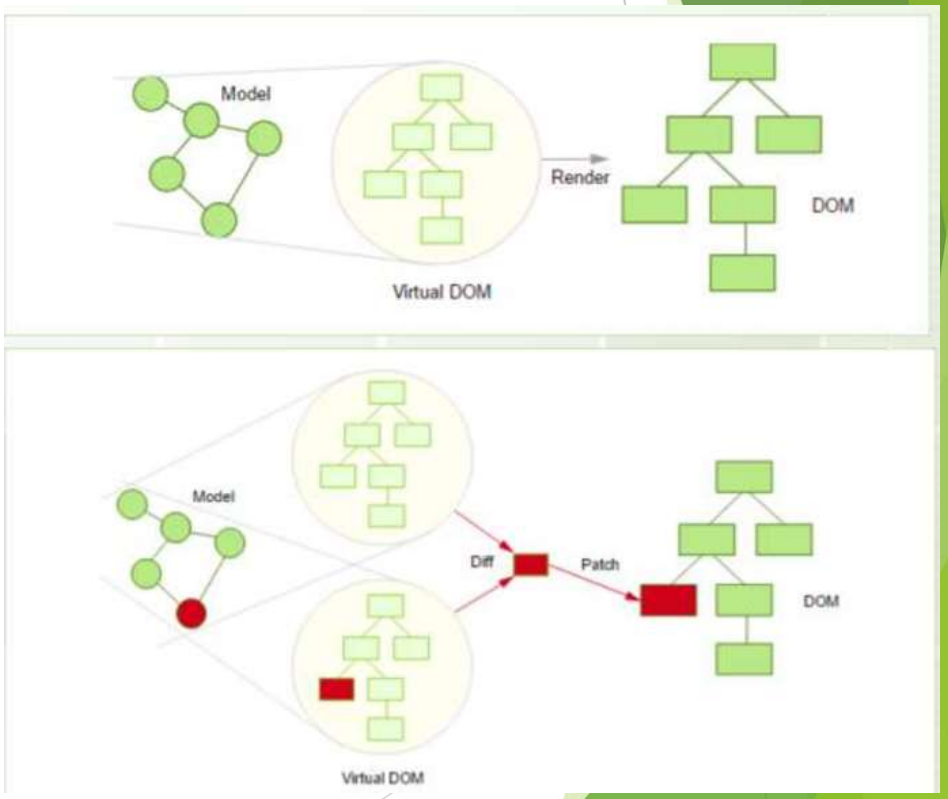


C’est quoi React JS

- React JS est une bibliothèque javascript développer par Facebook pour les applications Web
- React JS fournit une implémentation du ‘V’ dans une application basé sur une architecture MVC
- React JS utilise une architecture orientée composant
- React JS ne possède pas de controller, template, directive…
- React JS possède un Virutel DOM

C’est quoi Virtual DOM

- Le virtual DOM est une abstraction du DOM HTML
- Le but du virtual DOM est de répondre à certain problèmatique du DOM HTML
  - La manipulation du DOM est très couteuse en terme de ressource
  - Le rendering du DOM rend augmente la latence d’une application.
- En utilisant le DOM virtuel, react js identifie les changements et modifie que la partie concerné dans le DOM



Installation des outils de développement

- Un ide (visual studio code, atom…)
- Installation de node JS
- Installation du npm
- Installation du CRA

Javasript ES6 (ES2015)

- Introduction du Let
- Template et chaine de caractères
- Boucle For Of
- Fonction Lambda, ou Arrow Fonction
- Class et Héritge
- Exemples

Composant react JS

- Composant est l'unité de base dans tout application React js
- Un composant est :
  - Réutilisable
  - Maintenable
  - Testable
- Création d'un composant
  - A l'aide d'une simple fonction
  - A l'aite d'une classe
- Démo

Création d'un composant

- A l'aide d'une simple fonction

```
function BonjourComponent() {
    return React.createElement('p', {}, 'bonjour Component')
  }
  ReactDOM.render(
    React.createElement(BonjourComponent),
    document.getElementById('root')
  )
```

- A l'aide d'une classe

```
var Bonjour = React.createClass({
    render: function () {
        return React.createElement('p', {}, 'Bonjour tout le monde')
          }
        });
ReactDOM.render(<Hello/>, document.getElementById('Bonjour'));
```

Création d'un composant

- A l'aide d'une classe ES6

```
import React, { Component } from "react";

export class Test extends Component {

    render() {
        return (
            <div>
                Bonjour tout le monde
            </div>
        );
    }

}
```

Création d’un composant

- La classe React Component
- La methode Render
- Exemple de composan

JSX

- JSX est un langage de description du rendu des composant.
- JSX permet d'injecter directement un langage de description de balisage dans notre javascript

```
const name = 'Josh Perez';
const element = <h1>Hello, {name}</h1>;

ReactDOM.render(
  element,
  document.getElementById('root')
);
```

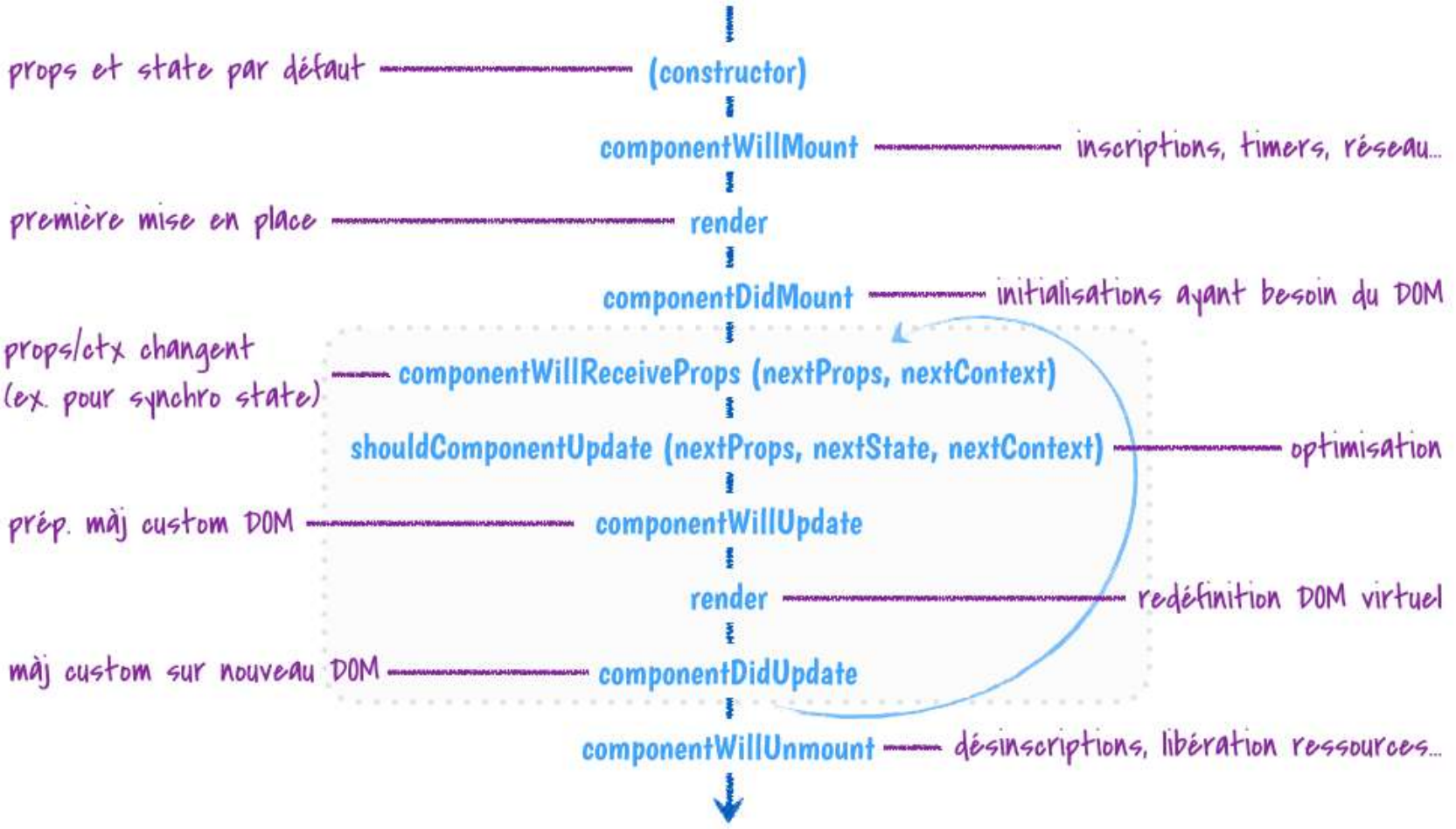
Les propriétés d’un composant

- les propriétés sont des informations accessibles par un composant initialisées lors de la création.
- Elles peuvent influer sur l’affichage du composant.
- On pourrait voir ces propriétés comme la configuration du composant
- En général un composant n’a jamais besoin de modifier ses propriétés
- Un composant n’a pas besoin de modifier ses propriétés, puisque les valeurs de celles-ci proviennent d’autres composants (généralement les composants parents).

L’etat d’un composant

- L’état représente les informations dont dépend l’affichage du composant (Comme les propriétés),
- Ces informations ne sont accessibles que par le composant lui même.
- Le composant va modifier son état en fonction d’évènements (abonnements/callback javascript, interactions utilisateurs…) qui lui sont extérieurs.
- Les mises à jours d’un composant se font à partir de mises à jour de l’état et non des propriétés

Cycle de vie d'un composant



Gestion des évènements

- React facilite énormément la gestion des événements du DOM, notamment en fournissant une syntaxe pratique et concise tout en conservant des performances optimales.
- Liste des événements

  - onChange

  - onClick

  -……..

Communication inter-composant

- D’un parent vers un enfant:
  - On utilise les propriétés pour passer des informations vers un composant enfant.
- D’un enfant vers un composant parent
  - On utilise les évènements pour passer les informations d’un composant enfant vers un composant parent.

Auto-Binding

- On peut binder des méthodes à notre jsx :
  - En utilisant pour chaque méthode la méthode bind de react js
  - En utilisant les expressions lambda
  - En utilisant des composant externe

Routing React js

- Utilisation du package react-router-dom
- Ajouter les routes
- Ajouter des paramètres dans les routes
- Redirection de lien

Test unitaire et test fonctionnel

- Principe des tests unitaire
- Utilisation de Jest et Jest snapshot
- Utilisation de react-test-renderer
- Limite des test unitaire
- Test fonctionnel
- Utilisation du mock jest.fn
- Utilisation de enzyme et enzyme-adapter-react-16

Ihab ABADI

Higher Order Components

- Higher order component est une fonction qui prend en paramètre un composant, et retourne un composant
- Higher order peut prendre plusieurs paramètre mais un seul composant.
- La fonction doit être une fonction purs sans effets de bords
- Création de composants réutilisable
- Exemple withrouter
- Bonne pratique

Render avec portals

- Render portals permets de rendre un composant enfant dans un DOM node extérieur(Avec createPortal)
- Les évènements de l'enfant sont toujours propagés vers le parent

Injection de dépendances avec Context

- React et context CreatContext
- Propagation des données d'un parent vers un petit enfant sans passer par l'enfant
- childContextTypes
- getChildContext
- Injection de dépendances et inversion de control

Fragments et render props

- Utilisation de Fragments
  - Fragment permet de rendre un groupe d'enfant sans ajouter un node DOM

- Utilisation de render props
  - Render props est une technique qui permet de propager une fonction à l'interieur d'une props

Redux

- Définition du pattern Redux
- Action
- Reducers
- Store
- Provider
- Fonction connect
- Utilisation des form avec redux from
- Appel asynchrone et utilisation de redux saga

Cycle de vie React

1 – Mounting

- constructor->render->componentDidMount

2- Updating

- shouldComponentUpdate->getSnapshotBeforeUpdate->componentDidUpdate

3- Unmounting

-componentWillUnmount

Composant purs et immutables

- Un composant purs est un composant écrit avec une fonction pure stateless sans effet de bord
- Les immutables ce sont des objets sui ont la particularité de conserver la même référence si leurs attributs n'ont subit aucun changement au cours d'une mise à jour.
- Utilisation de immutableJS