Ajuntament de Barcelona

Cibernàrium

Introducció a NodeJs

Capacitació Tecnològica
per a Professionals i Empreses

Formador:
Oriol Tinoco Marco
Formador &
CTO en Worktocloud, S.L.

Barcelonactiva

Barcelona Activa: Qui som?

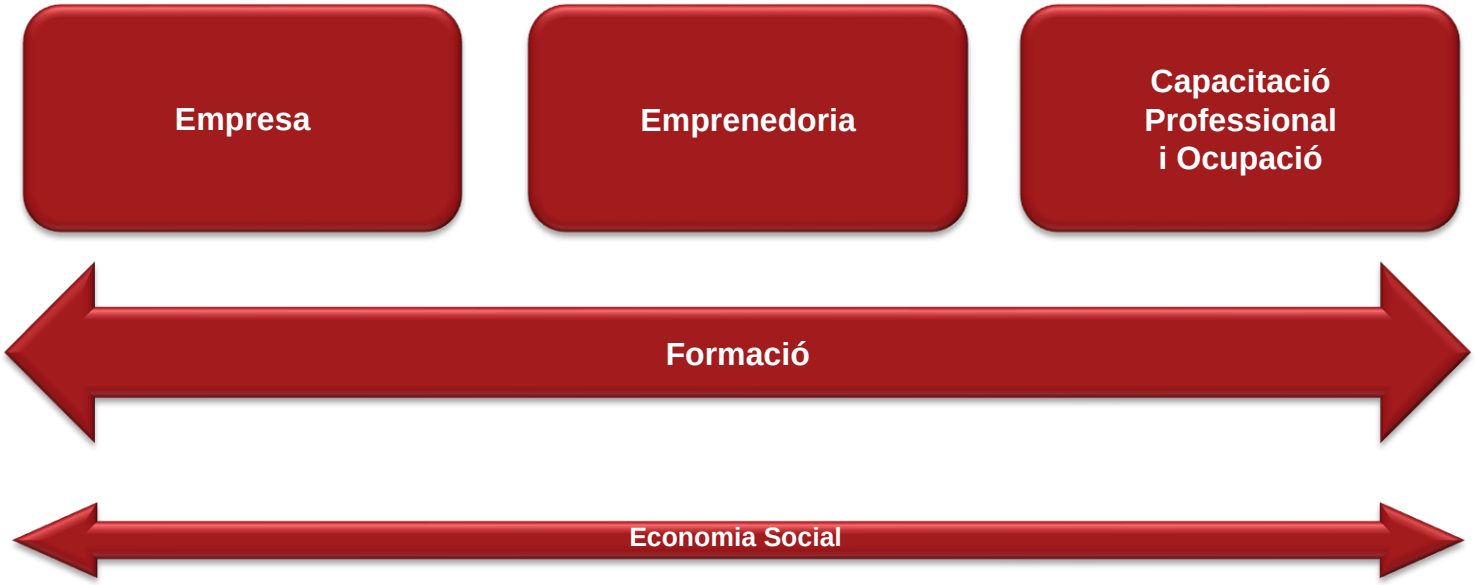
Barcelona Activa, integrada en l'àrea d'Economia, Empresa i Ocupació, és l'organització executora de les polítiques de promoció econòmica de l'Ajuntament de Barcelona.

Des de fa 25 anys impulsa el creixement econòmic de Barcelona i el seu àmbit d'influència donant suport a les empreses, la iniciativa emprenedora i l'ocupació, alhora que promociona la ciutat internacionalment i els seus sectors estratègics; en clau de proximitat al territori.

Barcelona Activa va ser guanyadora del Gran Premi del Jurat 2011, atorgat per la DG d'Empresa i Indústria de la Comissió Europea en el marc dels *European Enterprise Awards*, per la iniciativa empresarial més creativa i inspiradora d'Europa.

Àrees d'activitat de Barcelona Activa

Barcelona Activa s'estructura en tres grans blocs de serveis a les **Empreses**, a l'**Emprenedoria** i a la **Ocupació**. La **Formació** és un instrument transversal present en els tres blocs, així com també tot el relacionat amb l'economia social.



Una xarxa d'Equipaments Especialitzats

Xarxa de Proximitat

13 antenes Cibernàrium a biblioteques
10 punts d'atenció en Ocupació

5 Components

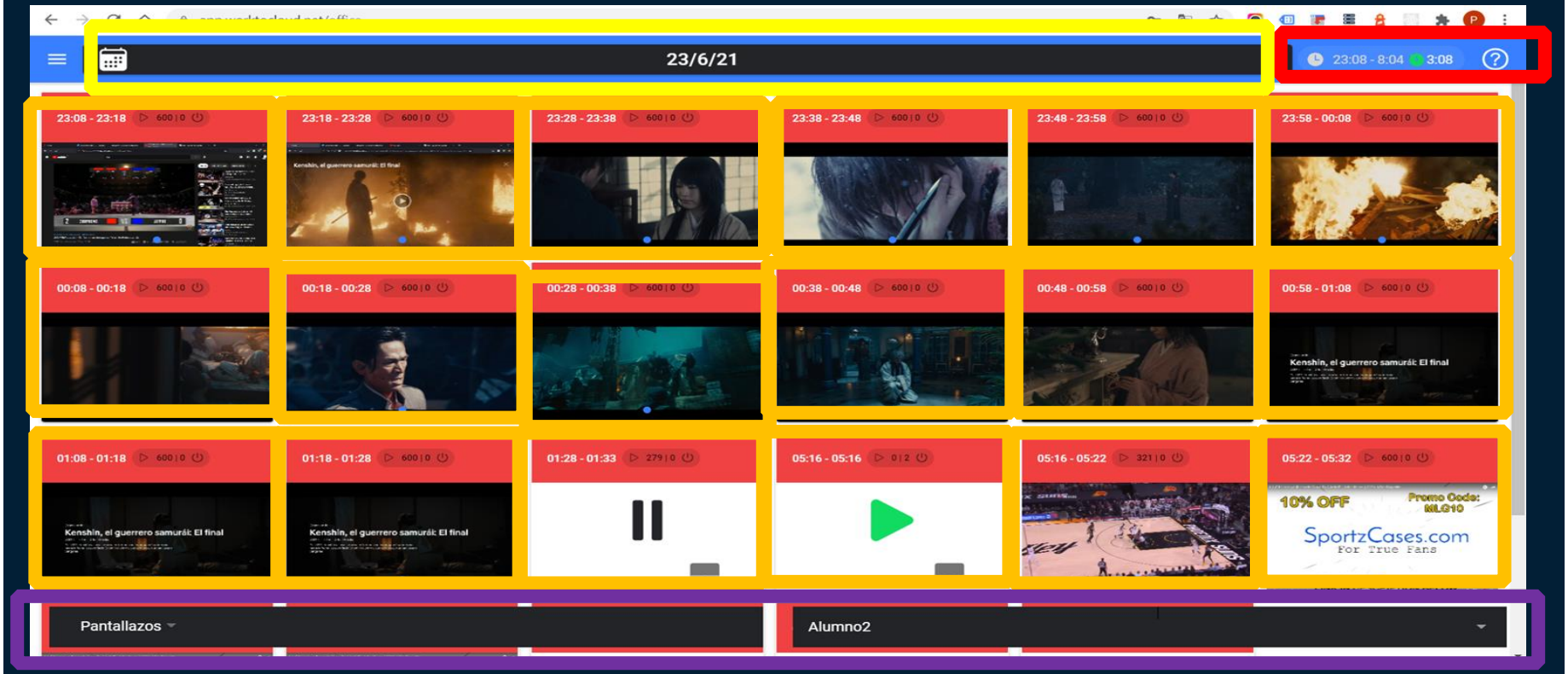
COMPONENTS

Els components de React són elements autònoms que podem reutilitzar a una pàgina. En crear petites peces de codi centrades, les podem moure i reutilitzar a mesura que la nostra aplicació s'amplia. La clau és que són autònomes i centrades, cosa que ens permet separar el codi en peces lògiques.

Motius per emprar components:

- Els components permeten separar la UI (Interfaç d'Usuari) en peces independents reutilitzables i pensar en cada component de forma a aïllada.
- Els 3 bigs frameworks / llibraries de JS (Angular, Vue i React) estan basats en components.

COMPONENTS


23/6/21
23:08 - 8:04 3:08
23:08 - 23:18
23:18 - 23:28
23:28 - 23:38
23:38 - 23:48
23:48 - 23:58
23:58 - 00:08
00:08 - 00:18
00:18 - 00:28
00:28 - 00:38
00:38 - 00:48
00:48 - 00:58
00:58 - 01:08
01:08 - 01:18
01:18 - 01:28
01:28 - 01:33
05:16 - 05:16
05:16 - 05:22
05:22 - 05:32
Kenshin, el guerrero samurái: El final
10% OFF
SportzCases.com
Pantallazos
Alumno2

Tipus de Components

Hi ha dos tipus de components personalitzats: basats en classes i funcionals. El primer component que crearem és un component basat en classes. Crearem un nou component anomenat Header que gestionarà l'encapçalament de la nostre app.

Nota: Els components basats en classes solien ser la manera més popular de crear components de React. Però amb la introducció dels hooks de React, molts desenvolupadors i biblioteques estan començant a fer servir components funcionals.

Si bé, ara, els components funcionals són la norma, sovint trobarem components de classes en codi heretat o antic. No cal que els usem, però síique sapiguem reconèixer-los. També ofereixen una introducció clara a molts conceptes futurs, com ara la gestió d'estats. En aquesta càpsula, aprendrem a crear components funcionals i de classe.

COMPONENTS DE CLASSE: CREACIÓ

Amb l'exemple del header mostrarem una imatge i un titol que podrem reutilitzar en totes les arees de la nostre app.

Per començar podem obtenir la imatge del Google Classroom en el Tema 5 i després desenvolupar el component. Recordem que encara no treballarem amb el disseny així que no té que quedar perfecte.

A continuació hem de crear un arxiu nou amb el nom de Header.js i així el·laborar el nostre primer component. Recordem que com a bona pràctica, tenim que escriure la primera lletra en majúscules.

Dins de l'arxiu hem de realizar el primer import:

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

La importació de React convertirà el JSX i alhora Component actuarà com a clase base que ampliarà els recursos propis de React per poder definir el nostre component.

COMPONENTS DE CLASSE: CREACIÓ

Per continuar amb la definició del nostre component de clase, permetre la importació del logo i desenvolupar l'estructura del component per poder ser emprada en la app, tenim que afegir les següents línies:

```
import logo from './qx7jYSYm_400x400.jpg';
class Header extends Component {

  render() {
    return(
      <h1><img src={logo} height="50px" /> El teu portal de montanya.</h1>
    )
  }

}

export default Header
```

No oblidem el afegir aquesta ultima linia per autoritzar

La clase Component de base te diversos mètodes que és poden emprar. En aquest cas el mètode render() retorna el còdi JSX que desitgem mostrar en el navegador.

COMPONENTS DE CLASSE: IMPLEMENTACIÓ

Ara ens cal implementar-ho en l'arxiu app.js important el Header i definint a on volem que és desplegui mitjantçant una etiqueta propia. Veiem el codi:

```
import './App.css';
import Header from './Header.js';

function App() {

  return (
    <>
      <Header/>
    </>
  );
}

export default App;
```

COMPONENTS FUNCIONALS: Creació

L'objectiu dels components funcionals, és el d'elaborar funcions de Javascript reaprofitables.

Per crear un component d'aquest tipus, hem de redactar una funció que habitualment pot ésser una funció fletxa i que per finalizar podem exportar com l'anterior. Veiem l'exemple que desenvoluparem:

```
import React from 'react';

function Menu({nick}){
    return (<span>Benvingut/da {nick} </span>);
}

export default Menu;
```

COMPONENTS FUNCIONALS: Implementació

Per poder implementar-ho correctamente, afegirem dins del component Header.js la crida del Salutacio.js i així poder enriquir l'encapçalament de la app. Podem veure com quedaría l'arxiu Header.js i tinguem en compte que l'App.js no tindrem que editar-lo.

```
import React, { Component } from 'react';
import logo from './qx7jYSYm_400x400.jpg';
import Salutacio from './Salutacio.js';
 class Header extends Component {
  render() {
    var usuari = "Oriol";
    return(
      <h1><img src={logo} height="50px" /> El teu portal de montanya.  <Salutacio nick={usuari} /> </h1>
    )
  }

}

export default Header
```

COMPONENTS DE CLASSE VS COMPONENTS FUNCIONALS

FUNCTIONAL COMPONENTS	CLASS COMPONENTS
Funció que accepta props com arguments i retorna react element.	Un class component ha de ser extended de React Component i retornar el React element dins del return del métode Render.
Stateless: (no te state) mostra les dades que rep com a props.	Statefull: te state i pot implementar lógica i state (x ex. carregar dades).
No te react **lifecycle** methods	Si té React **lifecycle** methods

Crear una estructura d'arxius llegible

En aquesta etapa, crearem una estructura d'arxius per organizar el seus components i recursos, com imatges, codi CSS i altres arxius de JavaScript. Agruparem el codi per components i alhora per tipus de recursos.

Recat és per defecte intencionalment independent de l'estructura d'arxius. No recomana ninguna estructura en particular i el projecte pot funcionar amb diverses jerarquies d'arxius. En canvi, recomano afegir cert ordre per evitar sobrecarregar el directori root (src) amb components, arxius CSS e imatges que seran dificils de navegar. A més d'emprar una nomenclatura explicita pot facilitar la visualització de les peces del nostre projecte que estan relacionades. Per exemple, pot no quedar clar que un arxiu de imatge anomenat Logo.jpg és part d'un component anomenat Header.js.

Així que inciarem per crear un directori anomenat components i a on situarem tots els components. A continuació crearem un directori img per les imatges.

COMPONENTS: Afegir Styling

Com recordareu a l'arxiu app.css hi ha els estils globals de la nostre app. En canvi, si només desitgem que s'apliquin a un component en concret, els crearem en un arxiu a part i d'espres importarem l'arxiu al component a on els volem fer servir. Veiem un exemple amb el Header.js

Header.css

```
.salutacio {
    color: rgb(24, 48, 6);
  }
```

COMPONENTS IMPORTAR ARXIU D'ESTILS LOCALS

```
import React, { Component } from 'react';
import './Header.css';
import logo from '../img/qx7jYSYm_400x400.jpg';
import Salutacio from './Salutacio.js';
 class Header extends Component {
  render() {
    var usuari = "Oriol";
    return(
      <h1 className="salutacio"><img src={logo} height="50px" /> El teu
portal de montanya.  <Salutacio nick={usuari} /> </h1>
    )
  }

}

export default Header
```

COMPONENTS: Resum

- Composició de componets: Aquests poden referirse a altres components en el seu procés de sortida. Això en permetrà emprar la mateixa abstracció per qualsevol nivel de detall (un botó, un quadre de diàleg, un formulari, una pantalla).
- Extracció de components: No tinguem por mai de dividir els components en d'altres de més petits.

REACT acostuma a ser molt flexible, però té una sola regla estricta:

Tots els components de React tenen que actuar com a funcions pures amb respecte a les seves propietats i d'aquesta manera mai tindrà que modificar les seves propietats.

6 Bootstrap

Gràcies!

Formador Oriol Tinoco

Linkedin: https://es.linkedin.com/in/oriol-tinoco-marco-589b9051

Email: orioltinoco@worktocloud.es

Barcelonactiva

Ajuntament de Barcelona

bcn.cat/barcelonactiva
bcn.cat/cibernarium