DE REACT 16.0 À REACT 16.8

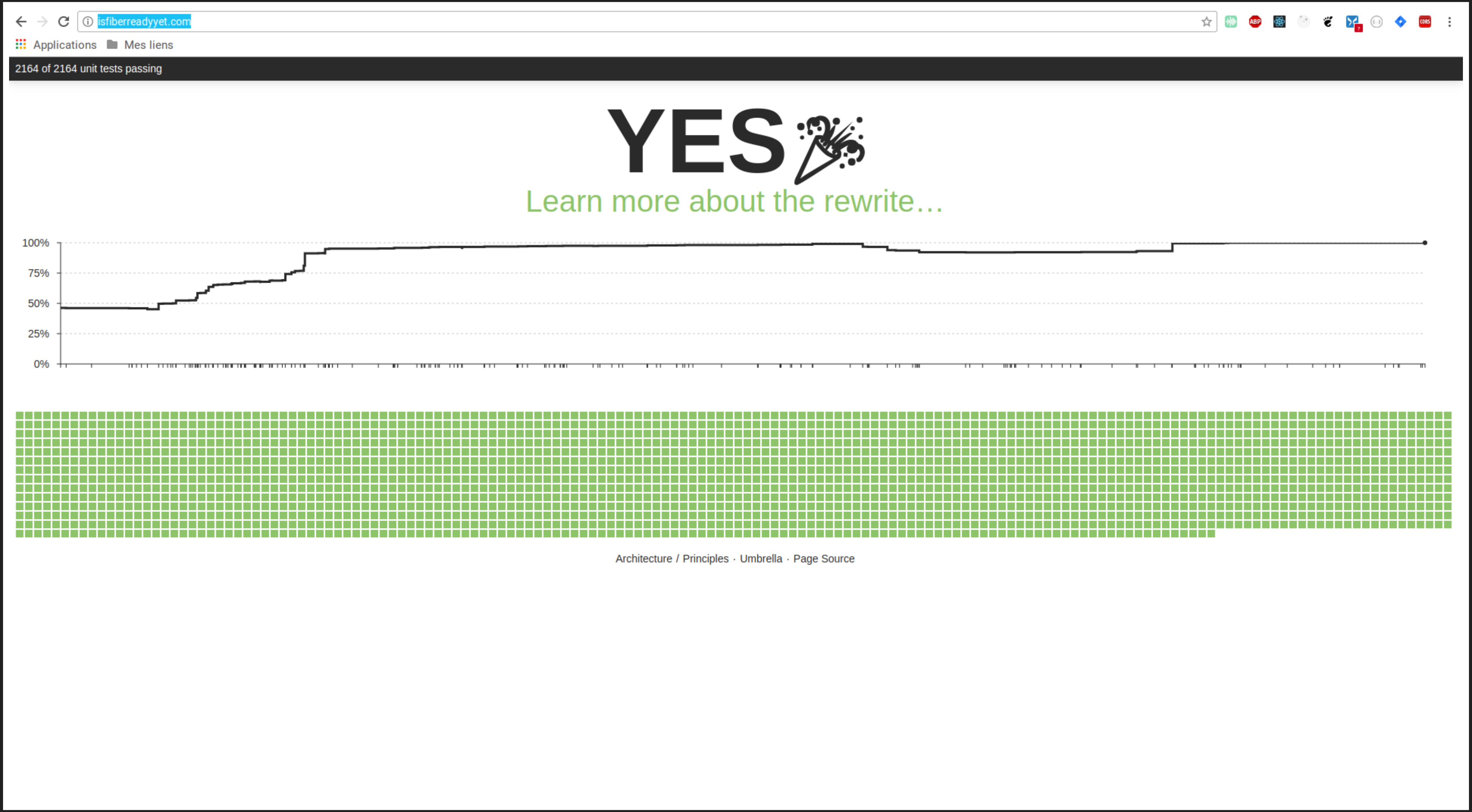
Sébastien Quenet
tech lead / coach agile @Abbeal

@Durnan (twitter)
github.com/SebQuenet

Il va parler des hooks tout de suite ?

(Moi je suis venu pour les hooks…)

IS FIBER READY YET ?



isfiberreadyyet.com

Applications Mes liens

2164 of 2164 unit tests passing

YES

Learn more about the rewrite…

Architecture / Principles · Umbrella · Page Source

REACT 15.6

(LA PRÉHISTOIRE)

CYCLE DE VIE DE REACT 15.6

INITIALISATION

```
class MyShinyComponent extends Component {

  componentWillMount() { ... }

  render() { ... }

  componentDidMount() { ... }

}
```

CYCLE DE VIE DE REACT 15.6

MODIFICATION

```
class MyShinyComponent extends Component {

  componentWillReceiveProps(nextProps) { ... }

  shouldComponentUpdate(nextProps, nextState) { ... }

  componentWillUpdate(nextProps) { ... }

  render() { ... }

  componentDidUpdate() { ... }

}
```

CYCLE DE VIE DE REACT 15.6

DESTRUCTION

```
class MyShinyComponent extends Component {

  componentWillUnmount() { ... }

}
```

TOUT VA BIEN NON ?

- Lent avec les animations
- DOM ultra verbeux
- Pas facile à débugguer
- Pas facile de faire des modales
- Grappes de propriétés qu'on doit propager à travers la hiérarchie
- React a besoin de libs externes
  - redux
  - recompose
  - loadable

REACT 16.0

26 SEPTEMBRE 2017

- Fragments
- Portals
- Error boundaries
- Async rendering
- Server-side rendering improvements

FRAGMENTS

```
render() {

          return (
          <>
            <div>Hello !</div>,
            <div>Hello world</div>,
          </>);

}
```

PORTALS

```
render() {

          return ReactDOM.createPortal(
            this.props.children,
            domNode,
          );

}
```

ERROR BOUNDARIES

```
class MyShinyComponent extends Component {

  componentWillReceiveProps(nextProps) { ... }

  shouldComponentUpdate(nextProps, nextState) { ... }

  componentWillUpdate(nextProps) { ... }

  render() { ... }

  componentDidUpdate() { ... }

  componentDidCatch(error) { ... }

}
```

ASYNC RENDERING

```
class MyShinyComponent extends Component {

  UNSAFE_componentWillMount() { ... }

  UNSAFE_componentWillReceiveProps() { ... }

  UNSAFE_componentWillUpdate() { ... }

}
```

SERVER-SIDE RENDERING

- Faster
- ReactDOM.hydrate()
- Streaming

Pardon mais c'est maintenant les hooks ?
C'est que j'ai peur qu'il ne reste pas assez de temps...

REACT 16.3

29 MARS 2018

- Strict Mode
- Context API
- React.createRef() & React.forwardRef()
- getDerivatedStateFromProps()
- getSnapshotBeforeUpdate()

STRICT MODE

```
<div>
  <React.StrictMode>
    <MyComponentTree />
  </React.StrictMode>
</div>
```

CONTEXT API

```
<OldGrandPa myLegacy={oldGrandPaLegacy}>
  <Daddy myLegacy={this.props.myLegacy}>
    <Myself myLegacy={this.props.myLegacy}>
      <MyChild myLegacy={this.props.myLegacy} />
    </MySelf>
  </Daddy>
</OldGrandPa>
```

CONTEXT API - PROVIDER

```
const LegacyContext = React.createContext([]);

<LegacyContext.Provider value={['pair of socks']}>
  <OldGrandPa>
    <Daddy>
      <Myself>
        <MyChild/>
      </MySelf>
    </Daddy>
  </OldGrandPa>
<LegacyContext.Provider>
```

CONTEXT API - CONTEXTTYPE (REACT 16.5)

```
class Myself extends React.Component {
  static contextType = LegacyContext;
  render() {
    return <MyChild myLegacy={context}>;
  }
}
```

CONTEXT API - CONSUMER

```
const LegacyContext = React.createContext([]);

<LegacyContext.Provider value={['pair of socks']}>

const myLegacyConsumer = () => (

  <LegacyContext.Consumer>
    { legacy => ( <div> {legacy[0]} </div> ) }
  </LegacyContext.Consumer>

)
```

REACT.CREATEREF()

```
class MyComponent extends React.Component {
  constructor(props) {
    super(props);
    this.inputRef = React.createRef();
  }

  render() {
    return <input type="text" ref={this.inputRef} />;
  }

  componentDidMount() {
    this.inputRef.current.focus();
  }
}
```

REACT.FORWARDREF()

```
const FancyButton = React.forwardRef((props, ref) => (
  <button ref={ref} className="FancyButton">
    {props.children}
  </button>
));

 const ref = React.createRef();
<FancyButton ref={ref}>Click me!</FancyButton>;
```

GET DERIVED STATE FROM PROPS

```
class MyShinyComponent extends Component {

  static getDerivedStateFromProps(props, state) { ... }

  shouldComponentUpdate(nextProps, nextState) { ... }

  render() { ... }

  componentDidMount/Update() { ... }

}
```

GET SNAPSHOT BEFORE UPDATE

```
class MyShinyComponent extends Component {

  static getDerivedStateFromProps(props, state) { ... }

  shouldComponentUpdate(nextProps, nextState) { ... }

  render() { ... }

  getSnapshotBeforeUpdate(prevProps, prevState) { ... }

  componentDidUpdate(prevProps, prevState, snapshot) { ... }

}
```

REACT 16.4

23 MAI 2018

- Pointer API

REACT 16.5

5 SEPTEMBRE 2018

- React profiler

REACT 16.6

23 OCTOBRE 2018

- getDerivedStateFromError
- React.memo()
- Code splitting

ERROR BOUNDARIES UPDATE

```
class MyShinyComponent extends Component {

  static getDerivedStateFromProps(props, state) { ... }

  shouldComponentUpdate(nextProps, nextState) { ... }

  render() { ... }

  componentDidUpdate() { ... }

  componentDidCatch(error) { ... }

  static getDerivedStateFromError() { ... }

}
```

REACT.MEMO()

```
const MyComponent = React.memo((props) => {
  /* only rerenders if props change */
});
```

CODE SPLITTING

```
import React, {
  lazy,
  Suspense,
 } from 'react';

const OtherComponent = lazy(
  () => import('./OtherComponent')
);

const MyComponent = () => (
  <Suspense fallback={<div>Loading...</div>}>
    <OtherComponent />
  </Suspense>
);
```

REACT 16.8

6 FÉVRIER 2019

HOOKS !

(c'est pas trop tôt … j'étais venu pour ça…)

USESTATE

```
const myComponent = newTitle => {
  const [ counter, setCounter ] = useEffect(0);
  setCounter(counter + 1);
  return(<div>{counter}</div>);
}
```

USEEFFECT

```
const myComponent = () => {
  const [ serviceData, setServiceData ] = useState(null);

  useEffect(
    () => {
      const serviceInstance = MyFancyService.connect();
      setServiceData( serviceInstance.fetchData() );
      return () => serviceInstance.disconnect();
    },
    []
  );
};
```

USECONTEXT

```
const ThemeContext = React.createContext({color: 'salmon'});

const myComponent = () => {

  const theme = useContext(ThemeContext);

  return (<p style={{ color: theme.color }}>
    Parce que, pourquoi pas ?
  </p>);
};
```

USEMEMO

```
const memoizedValue = useMemo(
  () => computeValue(a, b),
  [a, b]
);
```

USECALLBACK

```
const memoizedCallback = useCallback(() => {
  computeValue(a, b);
}, [a, b]);
```

USEREDUCER

```
const reducer = (state, action) => {
  switch(action.type) {
    case 'inc': return state + 1;
    case 'dec': return state - 1;
  }
};

const component = (initialState = 0) => {
  const [state, dispatch] = useReducer(reducer,initialState);
  return (<button onClick={() => dispatch({type: 'inc'})}/>);
};
```

OTHER BUILT-IN HOOKS

- useRef()
- useImperativeHandle()
- useLayoutEffect()
- useDebugValue()

SOURCES DES SLIDES TÉLÉCHARGEABLES ICI :

https://github.com/SebQuenet/React16.8-pres

MERCI !

DES QUESTIONS ?