

Sublime React ,es una de las necesidades que últimamente todo el mundo tiene. Poco a poco React.js esta cogiendo tracción como librería de capa de presentación y necesitamos integrarlo con uno de nuestros editores favoritos. Esto en principio no debería tener mayor misterio ya que React esta basado en JavaScript.

```
ReactDOM.render(  
  
  React.createElement("p",null,"hola react"),  
  document.getElementById("zona")  
);
```

El problema viene cuando usamos React.js a través de de JSX (JavaScript Sintax Extension) . En este caso el código de JavaScript es diferentes ya que usamos esta sintaxis peculiar que incluye XML.

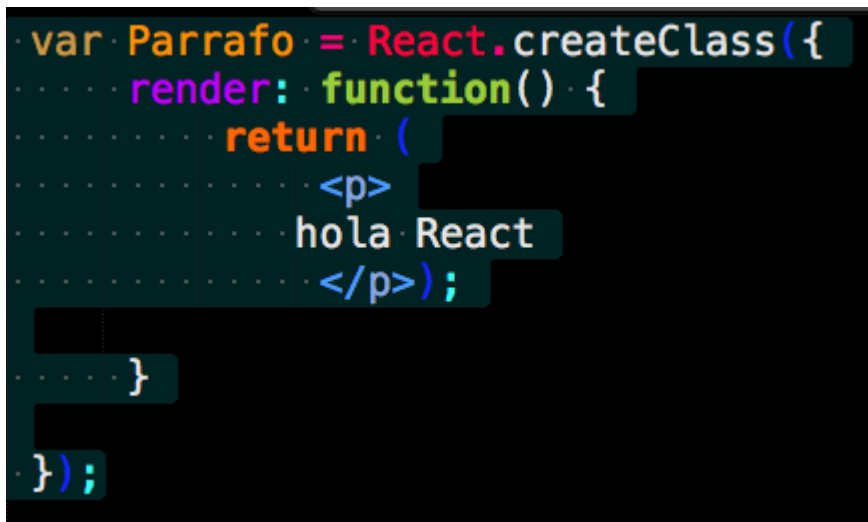
```
var Parrafo = React.createClass({  
  render: function() {  
    return (  
      <p>  
        hola React  
      </p>);  
  }  
});
```

El problema surge sobre cuales son los plugins adecuados a utilizar en Sublime para trabajar de forma cómoda con JSX.

Sublime React y sus problemas

Normalmente tenemos instalado [HTML-Prettify](#) como plugin de Sublime que nos ayuda a maquetar el código. Nada más empezamos a trabajar con JSX nos daremos cuenta que tenemos dos problemas . El primero, el syntax highlighting de Sublime no es suficiente y se vuelve loco a la hora de mostrarnos el código de JSX. En segundo lugar la maquetación de HTML-Prettify no es capaz de encajar la sintaxis de JSX, vamos a ver como solucionar estos problemas .

El primer paso es instalar el [plugin de Babel para Sublime](#) que incluye soporte para los ficheros JSX. Instalado este plugin dispondremos de nuevas opciones de sintaxis highlight para estos ficheros.

A screenshot of a code editor showing JSX code with syntax highlighting. The code is:

```
var Parrafo = React.createClass({
  render: function() {
    return (
      <p>
        hola React
      </p>
    );
  }
});
```

El código queda mucho más claro y encaja mucho mejor con lo que es JSX. Sin embargo HTML-Prettify no sera capaz de maquetarlo ya que por defecto no soporta este tipo de extensiones. Para solventar este problema deberemos editar el fichero de configuración de HTML-Prettify y añadir las extensiones de JSX. Para hacer esto deberemos ir al menú de Preferences>Package-Settings>HTML/CSS/JS Prettify en el apartado de js añadiremos:

```
"js": {
```

"allowed_file_extensions": ["js", "json", "jshintrc", "jsbeautifyrc", "jsx"],

Ya podemos maquetar los ficheros jsx pero el resultado es decepcionante:

```
var Parrafo = React.createClass({
  render: function() {
    return ( < p >
      hola React < /p > );
  }
});

var Parrafo2 = React.createClass({
  render: function() {
    return (
      <p>
        hola2 React
      </p>
    );
  }
});
```

La maquetación sigue generando problemas. Este se debe a que el plugin no es capaz de gestionar las etiquetas xml que tenemos en nuestro código. Debemos de volver a modificar otra parte del plugin y activar que no modifique los bloques xml.

"e4x": true, // Pass E4X xml literals through untouched

Ahora si podremos trabajar de una forma cómoda

```
var Parrafo = React.createClass({
  render: function() {
    return (
      <p>
        hola React
      </p>);
  }
});

var Parrafo2 =
  React.createClass({
    render: function() {
      return (
        <p>
          hola2 React
        </p>
      );
    }
  });
```

Acabamos de configurar Sublime React para que trabaje de forma natural con los ficheros JSX.

Otros artículos relacionados: [Introducción a React.js](#) , [Rx.js](#) y [programación Reactiva](#)