# Resumen Conceptual de los Primeros Pasos en React

**\*\*Configuración Inicial\*\***El primer paso para trabajar con React es crear una carpeta en un lugar accesible, como el escritorio, para evitar la necesidad de permisos de administrador. Dentro de esta carpeta, se crea un subdirectorio nombrado con una numeración para facilitar el orden de aplicaciones.  
\*Ejemplo:\* Crear una carpeta llamada '01-Intro-React' en el escritorio.

**\*\*Creación del archivo HTML\*\***Crear un archivo `index.html` en el editor de código, agregar un 'cascarón' básico de HTML y configurar el título como `React App`.  
\*Ejemplo:\* Utilizar el símbolo '!' y 'tab' en Visual Studio Code para generar la estructura base.

**\*\*Importación de Librerías React\*\***Agregar las librerías React, ReactDOM y Babel en el archivo HTML para que el navegador soporte el uso de código JSX, que es una mezcla de JavaScript y XML, permitiendo trabajar de forma óptima con React.  
\*Ejemplo:\* Incluir las librerías justo debajo de las etiquetas `<meta>`.

**\*\*Creación de Elementos JSX y Uso de Babel\*\***Crear una etiqueta `h1` en JSX con un texto, como `Hola mundo`, que debe ser procesado por Babel para que sea compatible con los navegadores. Configurar el script con el tipo `text/babel`.  
\*Ejemplo:\* Crear una constante `h1Tag` con un elemento `h1` y configurar el script con `type='text/babel'`.

**\*\*Renderizado en React\*\***Para visualizar el contenido de React en el navegador, se crea un `div` con un `id` de 'root' y se utiliza ReactDOM para renderizar los elementos dentro de este contenedor.  
\*Ejemplo:\* Utilizar `ReactDOM.render(h1Tag, divRoot)` para insertar el contenido en el elemento con `id='root'`.

**\*\*Insertar Variables en JSX\*\***Utilizar llaves `{}` para insertar variables dentro de elementos JSX, permitiendo el uso dinámico de valores.  
\*Ejemplo:\* Crear una constante `nombre` y colocar `{nombre}` dentro del `h1` para que muestre un saludo personalizado.

**\*\*Breve Introducción a Babel\*\***Babel es una herramienta que convierte el código moderno en un formato compatible con la mayoría de los navegadores. Para React, Babel permite que JSX sea comprendido por los navegadores de manera efectiva.  
\*Ejemplo:\* Utilizar `type='text/babel'` en los scripts de JSX para activar la conversión con Babel.