**React**

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called “components”.

**React** (старі назви: **React.js**, **ReactJS**) — [відкрита](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [JavaScript](https://uk.wikipedia.org/wiki/JavaScript" \o "JavaScript) [бібліотека](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC) для створення [інтерфейсів користувача](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), яка покликана вирішувати проблеми часткового оновлення вмісту [веб-сторінки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%B0), з якими стикаються в розробці [односторінкових застосунків](https://uk.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D1%81%D1%82%D0%BE%D1%80%D1%96%D0%BD%D0%BA%D0%BE%D0%B2%D0%B8%D0%B9_%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA" \o "Односторінковий застосунок). Розробляється [Facebook](https://uk.wikipedia.org/wiki/Facebook" \o "Facebook), [Instagram](https://uk.wikipedia.org/wiki/Instagram" \o "Instagram) і спільнотою індивідуальних розробників.

Онлайн редактор:

<https://codepen.io/anon/pen/BvRmrw>

========================================================================

**React “Hello world”**

[**https://reactjs.org**](https://reactjs.org)

Для роботи з React у браузері необідно підклчюити дві бібліотеки:

* React (бібліотека для побудови компонентів);
* ReactDOM (для формування інтерфейсу користувача).

Використання онлайн бібліотек (одна сторінка)

<https://raw.githubusercontent.com/reactjs/reactjs.org/master/static/html/single-file-example.html>

Остання версія

|  |  |
| --- | --- |
|  |  |
| <!DOCTYPE html>  <html> | <!DOCTYPE html>  <html> |
| <head>  **. . . . . . . . . . . . . . . . . .**  **-------- Підключення бібліотек ---------------**  <script src="https://unpkg.com/react@16/umd/react.development.js"></script>  <script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>    <!-- Don't use this in production: -->  <script src="https://unpkg.com/babel-standalone@6.15.0/babel.min.js"></script>  **. . . . . . . . . . . . . . . . . .**  </head> | <head>  <meta charset="UTF-8" />  <title>Hello World</title>  **-------- Підключення бібліотек ---------------**  **<script src="https://unpkg.com/react@16/umd/react.development.js"></script>**  **<script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>**    **<!-- Don't use this in production: -->**  **<script src="https://unpkg.com/babel-standalone@6.15.0/babel.min.js"></script>**  </head> |
| <body>  **--------- Визначення місця для вставки контенту -------------**  **<div id="root"></div>** | <body>  **--------- Визначення місця для вставки контенту -------------**  **<div id="root"></div>** |
| **---------------- Генерування контенту -------------**  <script type="text/babel">  **ReactDOM.render**(  що\_вставляємо ,  куди  );  </script>  </body>  </html> | **---------------- Генерування контенту -------------**  <script type="text/babel">  **ReactDOM.render(**  **<h1>Hello, world!</h1>,**  **document.getElementById('root')**  **);**  </script>  </body>  </html> |

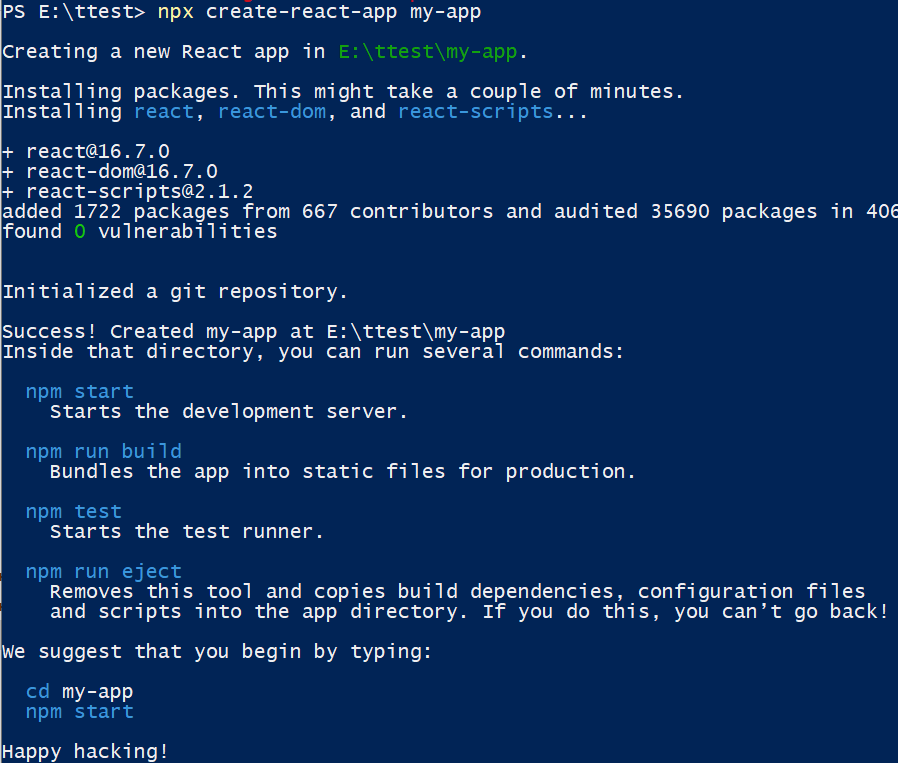
**Вставка звичайного JS**

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title>Pure React Samples</title>  </head>  <body>  <!-- Контейнер -->  **<div id="react-container"></div>**  <!-- Бібліотеки React та ReactDOM -->  <script src="https://unpkg.com/react@15.4.2/dist/react.js"></script>  <script src="https://unpkg.com/react-dom@15.4.2/dist/react-dom.js"></script>  <script>  **ReactDOM.render(**  **React.DOM.h1(**  **{id: "my-heading"},**  **React.DOM.span(null,"Hello"),**  **" World!"**  **),**  **document.getElementById('react-container')**  **);**  </script>  </body>  </html> |

**Шлях з використанням консолі**

[**https://github.com/facebook/create-react-app**](https://github.com/facebook/create-react-app)



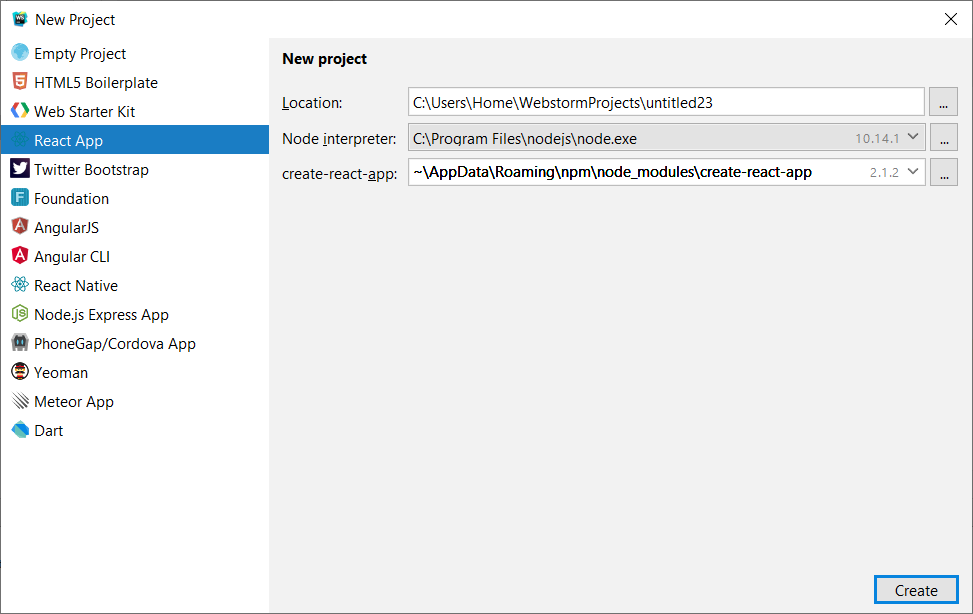


========================================================================

========================================================================

Встановлення в WebStorm

**npm i create-react-app -g**



Запуск додатку

**npm start**

========================================================================

========================================================================

**Вставка компонентів**

<https://babeljs.io/en/repl.html#?babili=false&browsers=&build=&builtIns=false&spec=false&loose=false&code_lz=GYVwdgxgLglg9mABMAFASgFAG8OMQJwFMoR8ldEAeAExgDcA-CvSgCwEZEZqBeAImAx8AZygBJanyZ4ZeABKEANorjMqAeg7SNtRhQwBfDBjadu_QSKhSFyuJU3sGQA&debug=false&forceAllTransforms=false&shippedProposals=false&circleciRepo=&evaluate=false&fileSize=false&timeTravel=false&sourceType=module&lineWrap=true&presets=es2015%2Creact%2Cstage-2&prettier=false&targets=&version=6.26.0&envVersion>=

**Способи створення екземплярів компонентів**

|  |  |  |
| --- | --- | --- |
| З використанням React.createElement | Використання методів-фабрик. | З використанням **JSX** |
| React.createElement(  компонент,  властивості,  ***дочірні*\_*елементи***  ); | **React.DOM. *метод-фабрика***  **(**  ***властивості* ,**    ***дочірні*\_*елементи***  **)** | <*компонент* властивості >  . . . дочірні\_елементи . . .  </*компонент* > |
| React.createElement(  "h1",  { id: "first" },  "Hello"  ); | React.DOM.h1(  {  id:"firstId"  },    "hello"  ) | <h1 id="first">  Hello  </h1> |

|  |  |
| --- | --- |
| Загальна форма | Приклад |
| React.createElement(  компонент,  {  властивість1 : значення1 ,  властивість2 : значення2 ,  . . . . .  },  ***дочірні*\_*елементи***  ); | React.createElement(  "h1",  { id: "first" },  "Hello"  ); |
| **React.DOM. *метод-фабрика***  **(**  {  властивість1 : значення1 ,  властивість2 : значення2 ,  . . . . .  }**,**    ***дочірні*\_*елементи***  **)** | React.DOM.h1(  {  id:"firstId"  },    "hello"  ) |
| <*компонент*  властивість1 = значення1  властивість2 = значення2  . . . .  >  . . . дочірні\_елементи . . .  </*компонент* >  **Якщо значення не рядок, а вираз JS (константи, фасиви, функції, об’єкти …), то значення треба вказувати у фігурних дужках** | <h1 id="first">  Hello  </h1>  <input type="checkbox" defaultChecked**={false}** /> |

Зауваження:

|  |  |
| --- | --- |
| Замість атрибуту **class** потрібно використовувати **className** |  |
| Замість HTML атрибуту **for** потрібно використовувати **htmlFor** |
| При заданні стилів треба вказувати style не як рядок, а як об’єкт |  |

Приклад.

|  |  |  |
| --- | --- | --- |
|  | | |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8" />  <title>Hello World</title>  <script src="https://unpkg.com/react@15.4.2/dist/react.js"></script>  <script src="https://unpkg.com/react-dom@15.4.2/dist/react-dom.js"></script>    </head>  <body>  <div id="root"></div>  <script> | | |
| **//--- З використанням JS (**createElement**)**  ReactDOM.render(  React.createElement(  “h1”,  {id: "my-heading"},  React.DOM.span(null,"Hello"),  " World!"  ),    document.getElementById('root')  ); | **//-- З використанням JS (мет.-фабрика)**  ReactDOM.render(    React.DOM.h1(  {id: "my-heading"},  React.DOM.span(null,"Hello"),  " World!"  ),    document.getElementById('root')  ); | //--- З використанням JSX ----  ReactDOM.render(  <h1 id: "my-heading">  <span>"Hello"</span>  </h1>,    document.getElementById('root')  );  //-------- Приклад 2 ------------  var ToDoList = React.createClass({  displayName:"ToDoList",  render: function() {  return React.DOM.ul(null,  this.props.list.map((item,i)=>  React.createElement("li",{key:i},item)  )  );  }  });  ReactDOM.render(  **<ToDoList list={["one","two","three"]}/>**,  document.getElementById("root")  ); |
| </script>    </body>  </html> | | |

Приклад.

|  |
| --- |
| React.createElement("h1",  {id: "recipe-0", 'data-type': "title"},  "Baked Salmon"  ) |
|  |

Зауваження. При створенні дочірніх елементів з даних з масиву кожному елементу необхідно задати властивість key

React.createElement("ul", {className: "ingredients"},

items.map((ingredient, i) =>

React.createElement("li", **{ key: i }**, ingredient)

)

З використанням JSX

<**ul**>  
 {[**"one"**, **"two"**, **"three"**].map((item,i)=><**li key=**{i}>{item}</**li**>)}  
</**ul**>

**Створення власних компонентів класу**

|  |  |
| --- | --- |
| Загальний формат | Приклад |
| var назва\_компонента = React.createClass({  . . . . . . . . . . .  //---- Обов’язкова функція -----  render: function() {  . . . . . .  return компонент  }  }); | var MyComponent = React.createClass({  displayName:"MyCompp", //🡨 **Буде використовувати як тег**  render: function() {  return React.DOM.span(null, "Мій компонент");  }  });  //----------- Приклад 2  var ToDoList = React.createClass({  displayName:"ToDoList", //🡨 **Буде використовувати як тег**  render: function() {  return React.DOM.ul(null,  this.props.list.map((item,i)=>  React.createElement("li",{key:i},item)  )  );  }  });  ReactDOM.render(  React.createElement(  ToDoList,  {list:["one","two","three"]}  ),  document.getElementById("react-container")  ); |

Створення метода-фабрики

|  |  |
| --- | --- |
|  | **// ----- Опис компоненту -----**  var MyComponent = React.createClass({  displayName:"MyCompp",  render: function() {  return React.DOM.span(null, "Мій компонент");  }  }); |
|  | //---- Створення фабрики  var ComponentFactory = **React.createFactory**(Component);  //---- Використання фабрики  ReactDOM.render(  ComponentFactory(), //🡨 Використання фабрики  document.getElementById("app")  ); |

**Створення нового компонента як нащадка React.Component**

<https://reactjs.org/tutorial/tutorial.html#overview>

|  |  |
| --- | --- |
|  |  |
| class **нащадок** extends **React.Component** {  render() {  **. . . . . . . . . . . . . .**  **генерування контенту**  **. . . . . . . . . . . . . .**  }  } | **---- Приклад 1---**  var MyComponent = extends React.Component {  render: function() {  return React.DOM.span(null, "Мій компонент");  }  });  **---- Приклад 2 ---**  class ShoppingList extends React.Component {  **render()** {  return (  <div className="shopping-list">  <h1>Shopping List for {this.props.name}</h1>  <ul>  <li>Instagram</li>  <li>WhatsApp</li>  <li>Oculus</li>  </ul>  </div>  );  }  }  **---- Приклад 2 ---**  class Timer extends **React.Component** {  constructor(props) {  **super(props);**  this.state = { seconds: 0 };  }  tick() {  this.setState(state => ({  seconds: state.seconds + 1  }));  }  componentDidMount() {  this.interval = setInterval(() => this.tick(), 1000);  }  componentWillUnmount() {  clearInterval(this.interval);  }  **render() {**  return (  <div>  Seconds: {this.state.seconds}  </div>  );  }  }  ReactDOM.render(  <Timer />,  document.getElementById("root"));  -------Приклад 3 -----------------------------------------  class IngredientsList extends **React.Component** {  renderListItem(ingredient, i) {  return React.createElement("li", { key: i }, ingredient)  }  render() {  return React.createElement(  "ul",  {className: "ingredients"},  this.props.items.map(this.renderListItem)  )  }  } |

**Функціональні компоненти, що не зберігають стану. В цьому випадку замість класу описуємо функцію, яка повертає об’єкт компонента**

|  |  |
| --- | --- |
|  |  |
|  | const IngredientsList = ({items}) =>  React.createElement(  "ul",  {className: "ingredients"},  items.map((ingredient, i) =>  React.createElement("li", { key: i }, ingredient)  )  ) |

**Властивості**

При створенні об’єкта компонента можуть передаватися значення властивостей. Ці властивості доступні через властивість **props**

|  |  |
| --- | --- |
|  |  |
|  | <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title>Pure React Samples</title>  </head>  <body>  <!-- Целевой контейнер -->  <div id="react-container"></div>  <!-- Библиотеки React и ReactDOM -->  <script src="https://unpkg.com/react@15.4.2/dist/react.js"></script>  <script src="https://unpkg.com/react-dom@15.4.2/dist/react-dom.js"></script>  <script>  **var Component = React.createClass({**  **render: function() {**  **return React.DOM.span(null, "My name is " +**  **this.props.name); //🡨 Звертання до властивості name**  **}**  **});**  **ReactDOM.render(**  **React.createElement(Component, //🡨 Створення компонента**  **{**  **name: "Bob", //🡨 Задання властивості name**  **}),**  **document.getElementById("react-container")**  **);**  </script>  </body>  </html>  --------------------------------------- |

Задання специфікацій для властивостей (типу даних, обов’язковості, …) здійснюється за допомогою властивості propTypes

|  |  |
| --- | --- |
|  |  |
|  | <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title>Pure React Samples</title>  </head>  <body>  <!-- Целевой контейнер -->  <div id="react-container"></div>  <!-- Библиотеки React и ReactDOM -->  <script src="https://unpkg.com/react@15.4.2/dist/react.js"></script>  <script src="https://unpkg.com/react-dom@15.4.2/dist/react-dom.js"></script>  <script>  **var Component = React.createClass({**  **propTypes:{**  **name:React.PropTypes.string.isRequired**  **},**  **render: function() {**  **return React.DOM.span(null, "My name is " +**  **this.props.name);**  **}**  **});**  **ReactDOM.render(**  **React.createElement(**  **Component,**  **{**  **name: "Bob",**  **}),**  **document.getElementById("react-container")**  **);**  </script>  </body>  </html> |
| При роботі з класами властивість propTypes задається для класу (після опису класу | //---- Опис класу ------    **//---- Додавання властивості** |

Первірка типів



Задання обов’язковості передачі значень **isRequired**

|  |  |
| --- | --- |
|  |  |
|  |  |

**Задання значень за замовчуванням getDefaultProps**

|  |  |
| --- | --- |
|  |  |
|  |  |
| При створенні класів  значення за замовчуванням додаються як властивості класу (після опису класу) | //---- Опис класу ------    **//---- Додавання значень за замовчуванням** |

**Для класів властивості propTypes, defaultProps можна задавати як статичні поля**

|  |  |
| --- | --- |
|  |  |
|  |  |

**Настроювана перевірка властивостей (треба передати функцію, яка повертає помилку (якщо така є) або null, якщо помилки немає**

|  |  |
| --- | --- |
|  |  |
|  |  |

**Задання властивостей для функціональних компонентів без збереження стану**

|  |  |
| --- | --- |
|  |  |
|  | **-------------- Або ж задаємо значення за замовчуванням для функції -------------------------** |

**Робота з дочірніми елементами (властивість ref)**

|  |  |
| --- | --- |
|  |  |
|  |  |

======================================================================================================

Відслідковування стану (властивість **state**)

|  |  |
| --- | --- |
|  |  |
|  |  |