JavaScript Environment

JavaScript Runtime

When using React Native, you're going to be running your JavaScript code in up to three environments:

- In most cases, React Native will use Hermes, an open-source JavaScript engine optimized for React Native.
- If Hermes is disabled, React Native will use JavaScriptCore, the JavaScript engine that powers Safari. Note that on iOS, JavaScriptCore does not use JIT due to the absence of writable executable memory in iOS apps.
- When using Chrome debugging, all JavaScript code runs within Chrome itself, communicating with native code via WebSockets. Chrome uses V8 as its JavaScript engine.

While these environments are very similar, you may end up hitting some inconsistencies. It is best to avoid relying on specifics of any runtime.

JavaScript Syntax Transformers

Syntax transformers make writing code more enjoyable by allowing you to use new JavaScript syntax without having to wait for support on all interpreters.

React Native ships with the Babel JavaScript compiler. Check Babel documentation on its supported transformations for more details.

A full list of React Native's enabled transformations can be found in @react-native/babel-preset.

TRANSFORMATION	CODE
ECMAScript 5	
Reserved Words	

TRANSFORMATION	
	`promise.catch(function() {...});`
ECMAScript 2015 (ES6)	
Arrow functions	`<C onPress={() => this.setState({pressed: true})} />`
Block scoping	`let greeting = 'hi';`
Call spread	`Math.max(...array);`
Classes	`class C extends React.Component {render() { return <View />; }}`
Computed Properties	`const key = 'abc'; const obj = {[key]: 10};`
Constants	`const answer = 42;`
Destructuring	`const {isActive, style} = this.props;`
for...of	`for (var num of [1, 2, 3]) {...};`
Function Name	`let number = x => x;`
Literals	`const b = 0b11; const o = 0o7; const u = 'Hello\u{000A}\u{0009}!';`
Modules	`import React, {Component} from 'react';`
Object Concise Method	`const obj = {method() { return 10; }};`
Object Short Notation	`const name = 'vjeux'; const obj = {name};`
Parameters	`function test(x = 'hello', {a, b}, ...args) {}`

TRANSFORMATION	CODE
Rest Params	`function(type, ...args) {};`
Shorthand Properties	`const o = {a, b, c};`
Sticky Regex	`const a = /o+/y;`
Template Literals	`const who = 'world'; const str = `Hello ${who}`;`
Unicode Regex	`const string = 'foo💩bar'; const match = string.match(/foo(.)bar/u);`
ECMAScript 2016 (ES7)	
Exponentiation Operator	`let x = 10 ** 2;`
ECMAScript 2017 (ES8)	
Async Functions	`async function doStuffAsync() {const foo = await doOtherStuffAsync();};`
Function Trailing Comma	`function f(a, b, c,) {};`
ECMAScript 2018 (ES9)	
Object Spread	`const extended = {...obj, a: 10};`
ECMAScript 2019 (ES10)	
Optional Catch Binding	`try {throw 0; } catch { doSomethingWhichDoesNotCareAboutTheValueThrown();}`
ECMAScript 2020 (ES11)	
Dynamic Imports	`const package = await import('package'); package.function()`

TRANSFORMATION	CODE
Nullish Coalescing Operator	`const foo = object.foo ?? 'default';`
Optional Chaining	`const name = obj.user?.name;`
ECMAScript 2022 (ES13)	
Class Fields	`class Bork {static a = 'foo'; static b; x = 'bar'; y;}`
Stage 1 Proposal	
Export Default From	`export v from 'mod';`
Miscellaneous	
Babel Template	`template(`const %%importName%% = require(%%source%%);`);`
Flow	`function foo(x: ?number): string {};`
ESM to CJS	`export default 42;`
JSX	`<View style={{color: 'red'}} />`
Object Assign	`Object.assign(a, b);`
React Display Name	`const bar = createReactClass({});`
TypeScript	`function foo(x: {hello: true, target: 'react native!'}): string {};`

Polyfills

Many standard functions are also available on all the supported JavaScript runtimes.

Browser

- CommonJS `require`
- `console.{log, warn, error, info, trace, table, group, groupEnd}`
- `XMLHttpRequest , fetch`
- `{set, clear}{Timeout, Interval, Immediate}, {request, cancel}AnimationFrame`

ECMAScript 2015 (ES6)

- `Array.from`
- `Array.prototype.{find, findIndex}`
- `Object.assign`
- `String.prototype.{startsWith, endsWith, repeat, includes}`

ECMAScript 2016 (ES7)

- `Array.prototype.includes`

ECMAScript 2017 (ES8)

- `Object.{entries, values}`

Specific

- `__DEV__`

Is this page useful? 

Edit this page

*Last updated on **Aug 29, 2023***