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 <u>Hermes</u>, an open-source JavaScript engine optimized for React Native.
- If Hermes is disabled, React Native will use <u>JavaScriptCore</u>, the <u>JavaScript</u> engine that powers Safari. Note that on iOS, <u>JavaScriptCore</u> 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 <u>V8</u> 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 <u>Babel JavaScript compiler</u>. Check <u>Babel documentation</u> 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	<pre>promise.catch(function() {});</pre>	
ECMAScript 2015 (ES6)		
Arrow functions	<pre><c =="" onpress="{()"> this.setState({pressed: true})} /></c></pre>	
Block scoping	<pre>let greeting = 'hi';</pre>	
Call spread	<pre>Math.max(array);</pre>	
Classes	<pre>class C extends React.Component {render() { return <view></view>; }}</pre>	
Computed Properties	<pre>const key = 'abc'; const obj = {[key]: 10};</pre>	
Constants	const answer = 42;	
Destructuring	<pre>const {isActive, style} = this.props;</pre>	
forof	for (var num of [1, 2, 3]) {};	
Function Name	<pre>let number = x => x;</pre>	
Literals	<pre>const b = 0b11; const o = 0o7; const u = 'Hello\u{000A}\u{0009}!';</pre>	
Modules	<pre>import React, {Component} from 'react';</pre>	
Object Concise Method	<pre>const obj = {method() { return 10; }};</pre>	
Object Short Notation	<pre>const name = 'vjeux'; const obj = {name};</pre>	
Parameters	<pre>function test(x = 'hello', {a, b},args) {}</pre>	

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

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

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}

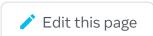
Specific

__DEV__

Is this page useful?







Last updated on Aug 29, 2023