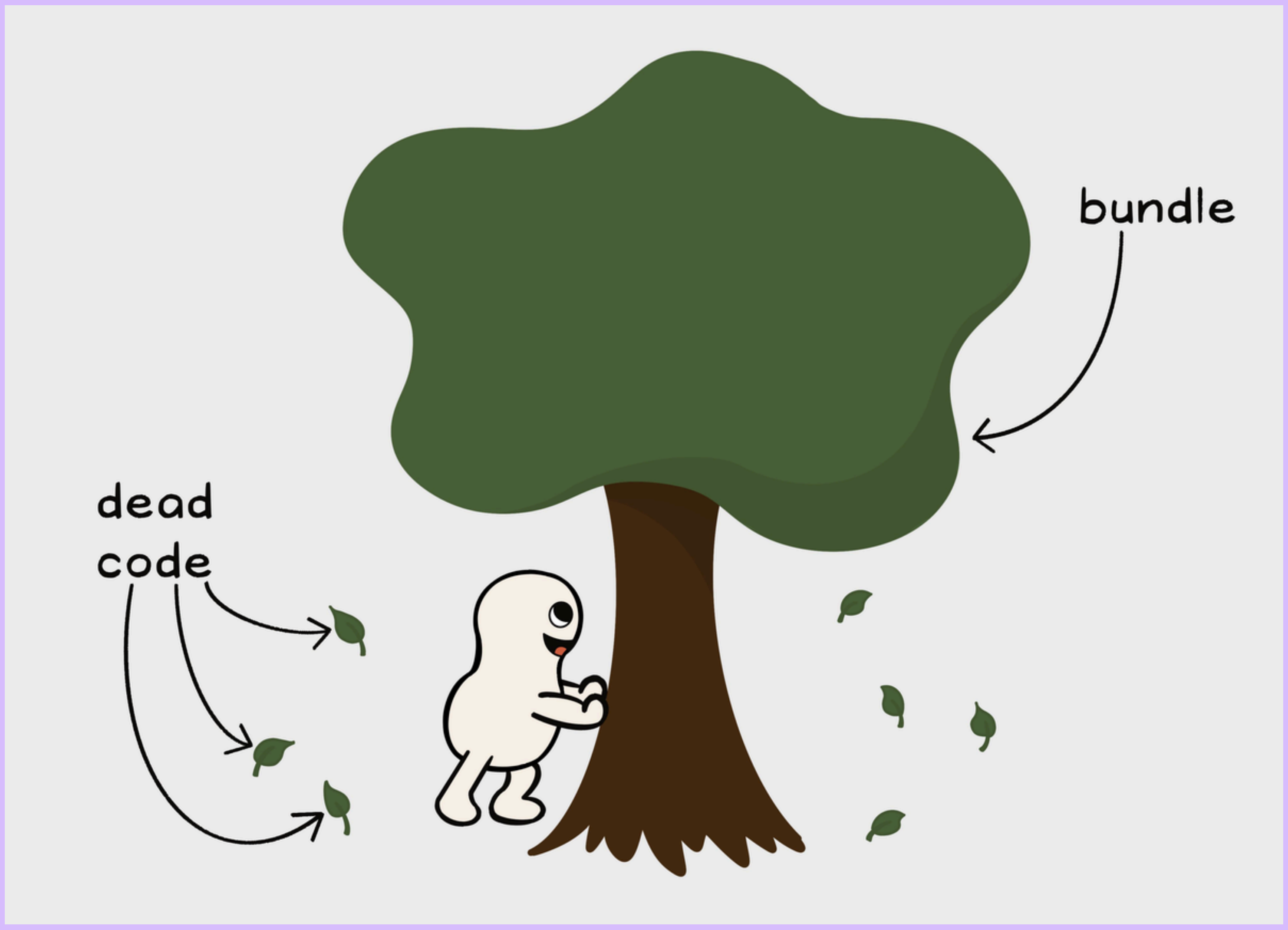
What is Tree Shaking in JavaScript

Vladyslav Demirov
@vladyslav-demirov

What is Tree Shaking?

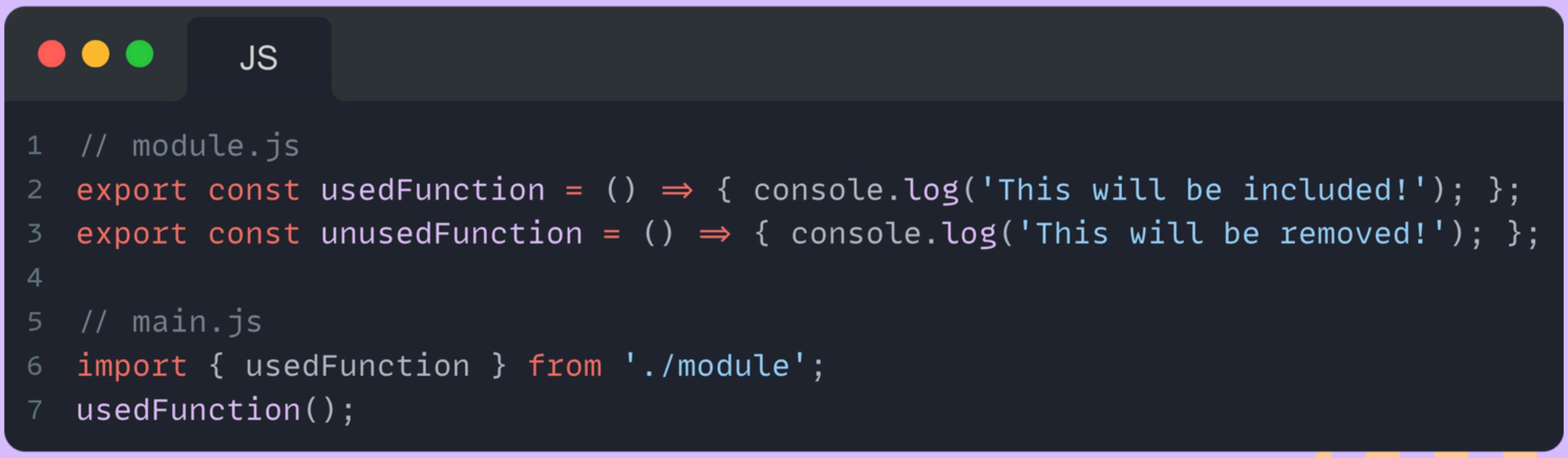
• Tree Shaking is a technique used in JavaScript bundlers like Webpack to eliminate dead code from your final bundle.
• By removing code that's never used, it reduces the bundle size and improves load times.
• Let's explore how it works and why it's essential for optimizing your applications.



Vladyslav Demirov

How Does Tree Shaking Work?

• Tree Shaking analyzes your import statements and only includes the code you actually use.
• It leverages ES6 module syntax (e.g., import and export) to identify and prune unused code.
• Tree Shaking is often performed by bundlers like Webpack, Rollup, or Parcel during the build process.



```js
// module.js
export const usedFunction = () => { console.log('This will be included!'); };
export const unusedFunction = () => { console.log('This will be removed!'); };

// main.js
import { usedFunction } from './module';
usedFunction();
```

In this example, only usedFunction will be included in the final bundle, while unusedFunction will be removed.

Vladyslav Demirov

Key Requirements for Tree Shaking

To take full advantage of Tree Shaking, you need:

1. ES6 Module Syntax: Use import and export statements.

2. Bundler Support: Tools like Webpack or Rollup are essential.

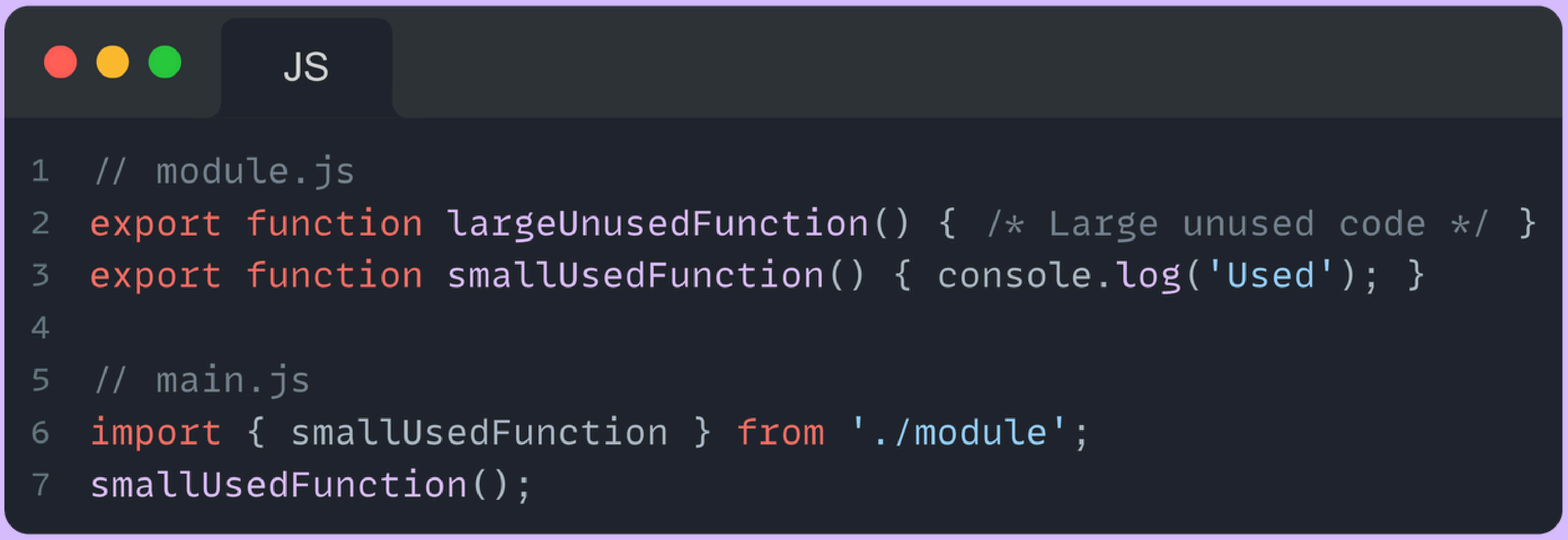
3. Pure Functions: Ensure functions don’t have side effects.

Vladyslav Demirov

Example

- Before Tree Shaking, all code from a module would be included in the final bundle, regardless of whether it was used.

- With Tree Shaking, only the necessary code is bundled.



```javascript
// module.js
export function largeUnusedFunction() { /* Large unused code */ }
export function smallUsedFunction() { console.log('Used'); }

// main.js
import { smallUsedFunction } from './module';
smallUsedFunction();
```

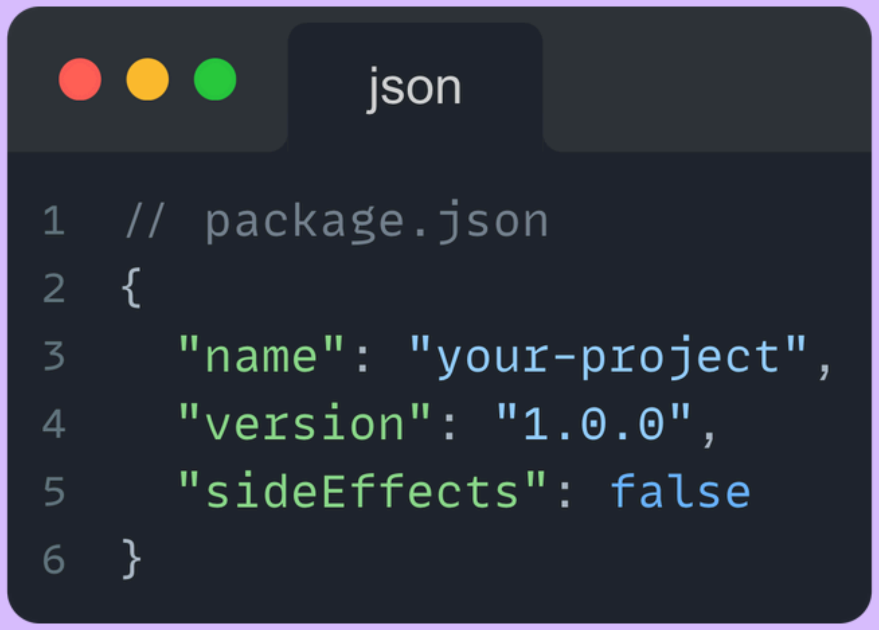
- After Tree Shaking, largeUnusedFunction is removed from the final bundle, resulting in a smaller, optimized output.

Vladyslav Demirov

How to Implement Tree Shaking

Using Webpack:

1. Ensure your project is using ES6 modules.
2. Configure Webpack with mode: 'production' to enable optimizations like Tree Shaking.
3. Use the sideEffects property in package.json to mark modules that can be safely shaken.



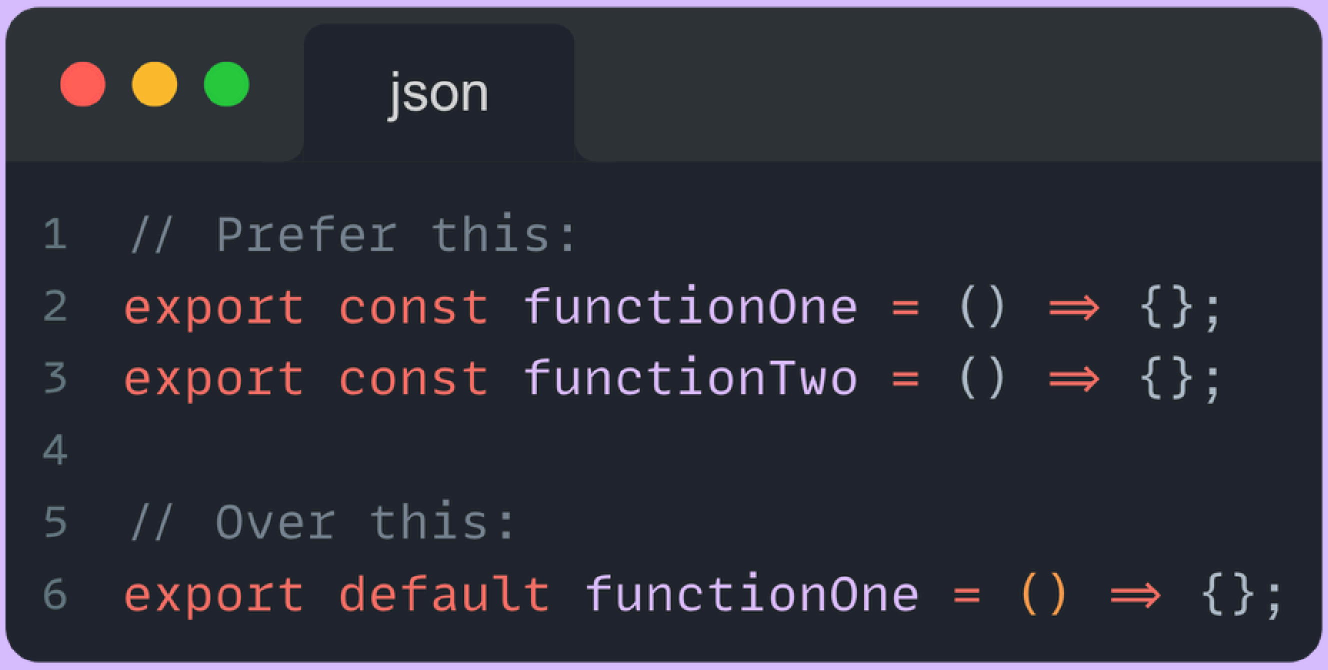
```json
// package.json
{
  "name": "your-project",
  "version": "1.0.0",
  "sideEffects": false
}
```

This setup tells Webpack that no files in your project have side effects, enabling more aggressive Tree Shaking.

Vladyslav Demirov

Advanced Tips for Tree Shaking

- Avoid Side Effects: Functions with side effects can prevent Tree Shaking from working effectively.
- Minify Your Code: Minification tools like Terser can work alongside Tree Shaking to further reduce bundle size.
- Use Named Exports: Prefer named exports over default exports, as they are more easily tree-shakable.



```
// Prefer this:
export const functionOne = () => {};
export const functionTwo = () => {};

// Over this:
export default functionOne = () => {};
```

Vladyslav Demirov

HAPPY

CODING

FOLLOW ME

Vladyslav Demirov

@vladyslav-demirov