BITS Pilani

Pilani Campus

Full Stack Application Development- SE ZG503

Akshaya Ganesan

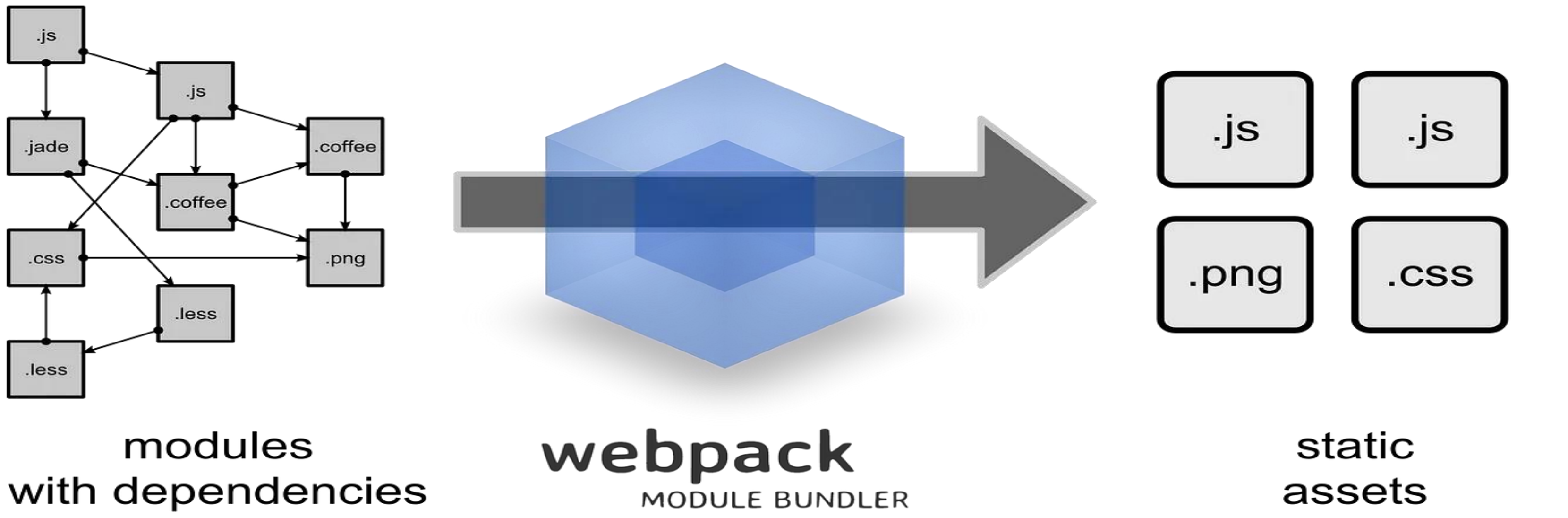
CSIS, WILP

BITS Pilani
Pilani Campus

Lecture No: 12.1 Webpack

Webpack

- Webpack is a robust and highly scalable open-source Javascript module bundler.
- Webpack is not limited to bundling javascript files and can bundle other file types such as CSS, SASS, images, typescript, etc.



Webpack- Dependency Graph

- To perform module bundling, Webpack conducts a process called dependency resolution
- Webpack looks through all the dependencies for the given modules and generates a **dependency graph**.
- A dependency graph represents how the modules are interdependent.
- Webpack uses this dependency graph to resolve dependency issues for the modules that depend on one another and generate one or more bundles.

Entry Point

- **Webpack configuration file:** A Webpack configuration file is a file that contains all the configurations for Webpack.
- The entry point contains the file path where Webpack begins its module bundling, and the entry point will act as the root to the dependency graph that Webpack generates. There can be multiple entry points, and a single entry point can also have multiple file paths passed as an array.
- Output: Output is the file location where Webpack stores the final bundled file.

Webpack execution

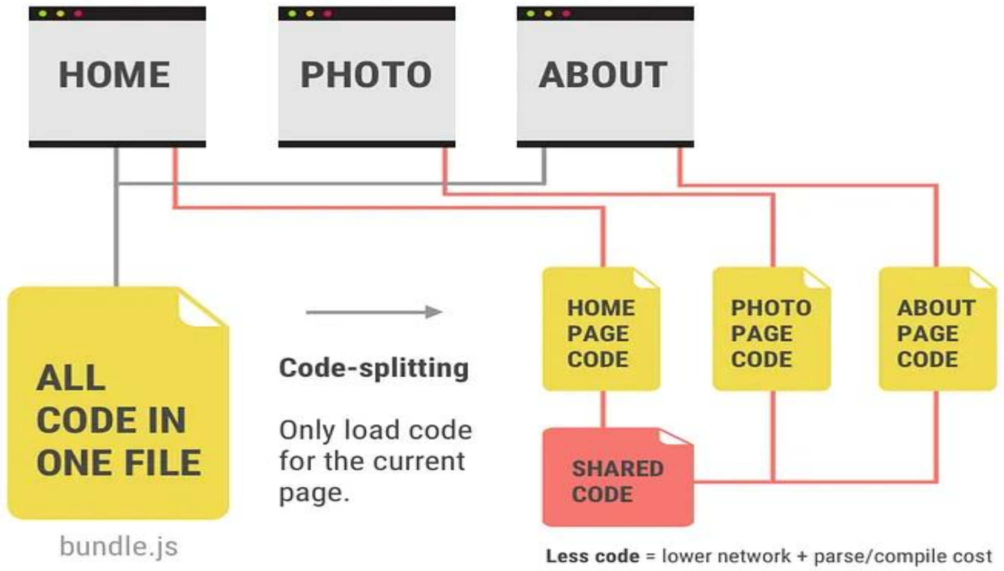
- When you run webpack, it:
  - Loads the webpack.config.js (or default config).
  - Webpack begins processing from the entry point.
  - Every time it sees an import or require, it treats it as a dependency and recursively parses those files.
  - As it parses, it builds a dependency graph
  - Every file it encounters is a module.
  - Webpack normalizes the module using its rules/loaders.
  - Loaders transform files from their original format to JavaScript that the bundler can include in the final bundle.

Webpack execution

- Webpack groups modules into chunks (e.g., one chunk per entry point or dynamic import).
- Each chunk contains all the modules needed for that part of the app.
- All chunks are stitched into a bundle (or multiple bundles).
- Writes the generated bundles and assets to the specified output folder.
- Optionally minifies or optimizes them (especially in production mode).
- Emits source maps if enabled.

Code splitting

- Code Splitting is an optimization technique.
- Code splitting is the process of splitting your project source code into multiple modules to be loaded in parallel or whenever needed.



Code Splitting

- Webpack identifies import() as a code split point.
- Webpack automatically extracts shared modules between chunks to avoid duplication.

Tree shaking

- In production mode: Webpack includes all exports in the output bundle but flags unused ones as **"unused export"**.
- **Tree shaking** is the process of **removing unused code** (a.k.a. “dead code”) from your final JavaScript bundle.
- Think of it like pruning branches from a tree: if a function or variable is **imported** but never used, Webpack (along with a minifier like Terser) will shake it out.

Naming Convention for output files

- Chunk name, usually based on your Webpack entry point:
- main.3f1c2.js
- **content hash** is a short hash string (e.g., MD5/SHA-256-based) generated from the file's content.
- If main.js changes, the hash changes → new filename.
- This helps in caching

sourcemap

- In production, your JavaScript is:
  - **Minified**: Remove whitespaces/comments, shorten longnames
  - **Bundled**: Many files combined into one
  - **Transpiled**: From newer syntax (ES6+, TypeScript) to older JS
- Debugging in the browser is difficult
- Source map: A file that maps minified code back to original source
- Source map : main.3a9c1.chunk.js.map

innovate achieve lead

BITS Pilani
Pilani Campus

Thank you