**Angular & Ivy Compiler**

**Ivy** is a complete rewrite of Angular’s rendering engine. In fact, it is the fourth rewrite of the engine and the third since Angular 2. But unlike rewrites two and three, which you might not have even noticed, Ivy promises huge improvements to your application. With Ivy, you can compile components more independently of each other. This improves development times since recompiling an application will only involve compiling the components that changed.

**Ivy** also has a very big focus on tree-shaking. This is the process in which the Typescript compiler looks at your code and figures out exactly which libraries are needed and then eliminates any unused code. As a result, the distributed code will be much smaller and the loading times of your application will improve.

**Ivy** has been around in a preview version since Angular 8, but you previously had to manually opt into using the new engine. With Angular 9, Ivy is the standard engine for rendering your content.

With all these massive changes behind the scenes, you might be scared and wonder how much you would need to refactor your code to be compatible with Ivy. It turns out that the Angular team has made backward compatibility a priority and, in most cases, you should not have to change anything in your application other than updating it to the latest Angular version.

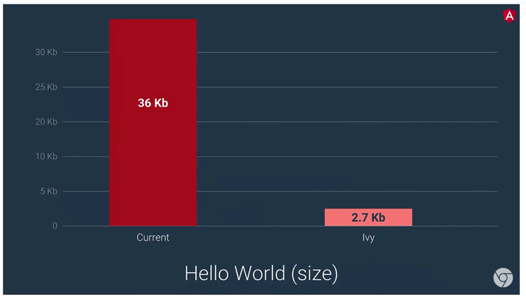
**Tree Shaking**

**Tree Shaking** is a term that means removing unused code during bundling process. This can be done by using tools like Rollup and Uglify. During the build process, tree shaking tools use static analysis and eliminates the unused and unreferenced code. However, tree shaking tools have limitations when the conditional code exists as static analysis depends on references. For example, an unused code path within “IF” statement cannot be identified by static analyzer and that code still resides in bundle even if it is not used during runtime. Due to the above mentioned limitations, current rendering pipeline is modified to optimize the bundle size.

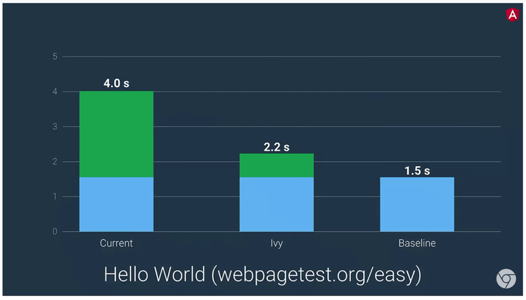
Tree Shaking is applied to Angular features. If any of the Angular features are not used in your code, you don’t need to pay for that code

**Test Results by Google**

Here are results of basic **Hello World** application provided by Google. Current angular generated **36Kb** size of bundle wherein Ivy generated **2.7Kb**, which is **93%** reduction in size.



**Google** tested the **Hello World** application on mobile devices that have slower network like **3G**. The current application load time is **4.0s** whereas **Ivy** application load time is **2.2s**, which is **45%** reduction in load time.



**Google** is still in verification process. Once it is done, **Google** will make **Ivy** a default renderer.

To start a new project with **Ivy**, use the **--enable-ivy** flag with the ng new command:

**npm i -g @angular/cli@latestng update**

**ng new angular-ivy-app --enable-ivy**

**To enable/update the existing project to Ivy Compiler:**

**npm i -g @angular/cli@latestng update**

OR in your **tsconfig.json** file update this:

**{**

**"compilerOptions": {**

**"module": "esnext",**

**// ...**

**},**

**"angularCompilerOptions": {**

**"enableIvy": true,**

**"allowEmptyCodegenFiles": true**

**}**

**}**

**angular.json:**

**{**

**"projects": {**

**"your-project": {**

**"architect": {**

**"build": {**

**"options": {**

**...**

**"aot": true,**

**}**

**}**

**}**

**}**

**}**

**}**

**Advantages of Ivy Compiler:**

1. **Smaller bundle sizes:**

The Ivy compiler has been designed to remove parts of Angular that aren’t being used via [tree-shaking](https://webpack.js.org/guides/tree-shaking/) and to generate less code for each Angular component.

1. **Faster testing:**

In Ivy, **TestBed** doesn’t recompile components between tests unless a component has been manually overridden, which allows it to avoid recompilation between the grand majority of tests. With this change, the framework’s core acceptance tests are about **40%** faster. We would expect users to see their own application test speeds to be around **40–50%** faster.

# **TestBed**

Configures and initializes environment for unit testing and provides methods for creating components and services in unit test

**Note:** **TestBed** is the primary API for writing unit tests for Angular applications and libraries. Use **TestBed** in tests. It will be set to either **TestBedViewEngine** or **TestBedRender3** according to the compiler used.

1. **Better debugging:**

Ivy provides you with more tools to debug your applications. When running an application in Dev Mode with the Ivy runtime, we now offer the new [ng object for debugging](https://v9.angular.io/api/core/global).

Ivy also improves the stack trace for debugging issues such as the ExpressionChangedAfterItHasBeenCheckedError.

1. **Improved CSS class and style binding:**

The Ivy compiler and runtime provides improvements for handling styles. Previously, if an application contained competing definitions for a style, those styles would destructively replace each other. With Ivy, the styles are merged in a predictable way.

1. **Improved type checking:**

The Angular compiler can check more of the types of your application, and it can apply more strict rules. These features will help you and your team catch bugs earlier in the development process.

* **fullTemplateTypeCheck** — Activating this flag tells the compiler to check everything within your template (ngIf, ngFor, ng-template, etc)
* **strictTemplates**— Activating this flag will apply the strictest Type System rules for type checking.

1. **Improved build errors:**

The Ivy compiler makes all of the error messages easier to read. This in turn helps a developer to resolve and fix the bugs at faster rate.

1. **Improved build times, enabling AOT on by default:**

For the first time ever we are using AOT even for DEV mode builds. This means that `ng serve` now benefits from the same compile-time checking as production builds, significantly improving the developer experience for Angular.

1. **Improved Internationalization:**

With Ivy, we’re making this faster by moving the build-time **i18n** substitutions later in the build process. This change allowed us to make it up to **10** times faster.

**Internationalization** is the process of designing and preparing your app to be usable in different languages. Localization is the process of translating your internationalized app into specific languages for particular locales.

Angular simplifies the following aspects of internationalization:

* Displaying dates, number, percentages, and currencies in a local format.
* Preparing text in component templates for translation.
* Handling plural forms of words.
* Handling alternative text.

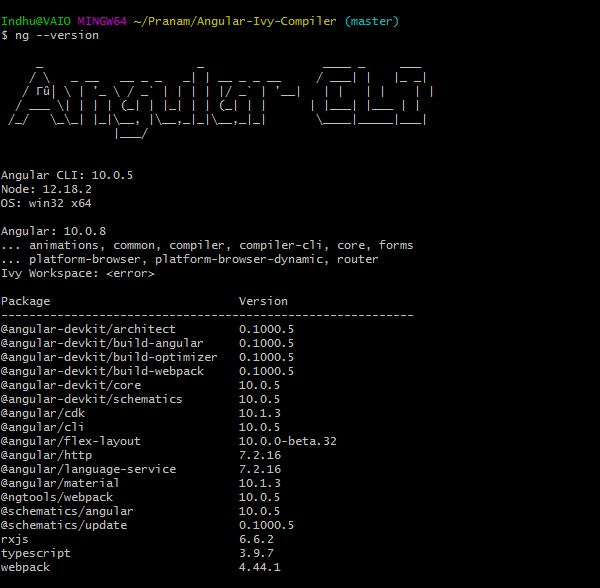
1. **Backward Compatibility:**

Ivy supports backward compatibility so that your existing applications wouldn't break. Today, thousands of applications are using angular framework. You don’t need to upgrade angular for Ivy.

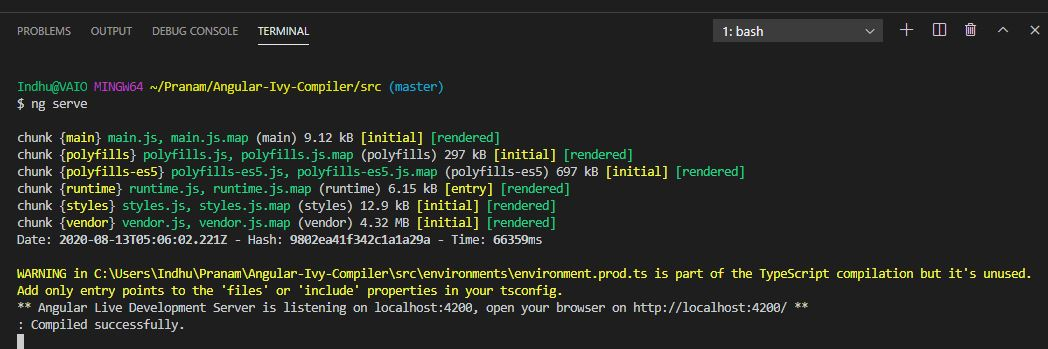
**Example:**

A sample Angular 10 project with Ivy Compiler:

Make sure you have the latest version of Angular:



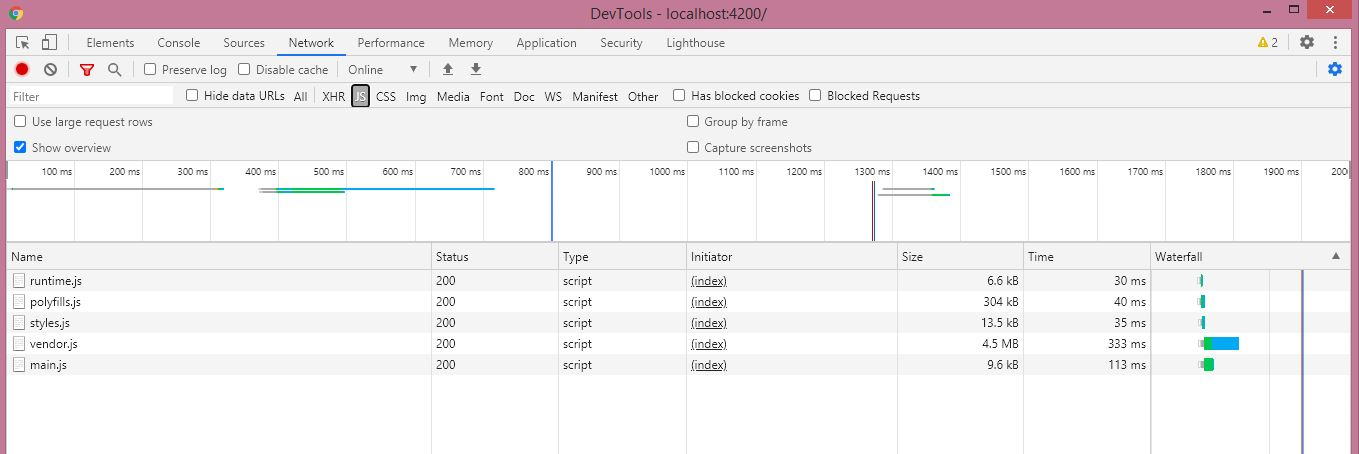
* 1. git clone <https://github.com/PranamBhat/Angular-Ivy-Compiler>
  2. cd Angular-Ivy-Compiler
  3. npm install
  4. npm start (or) ng serve

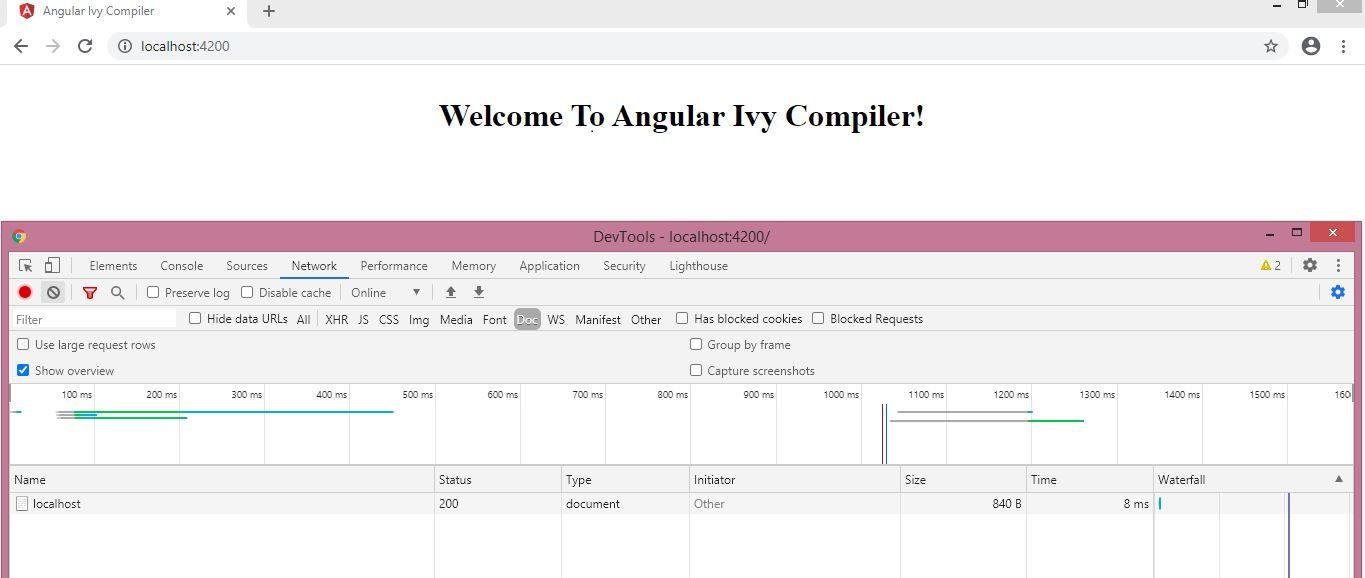


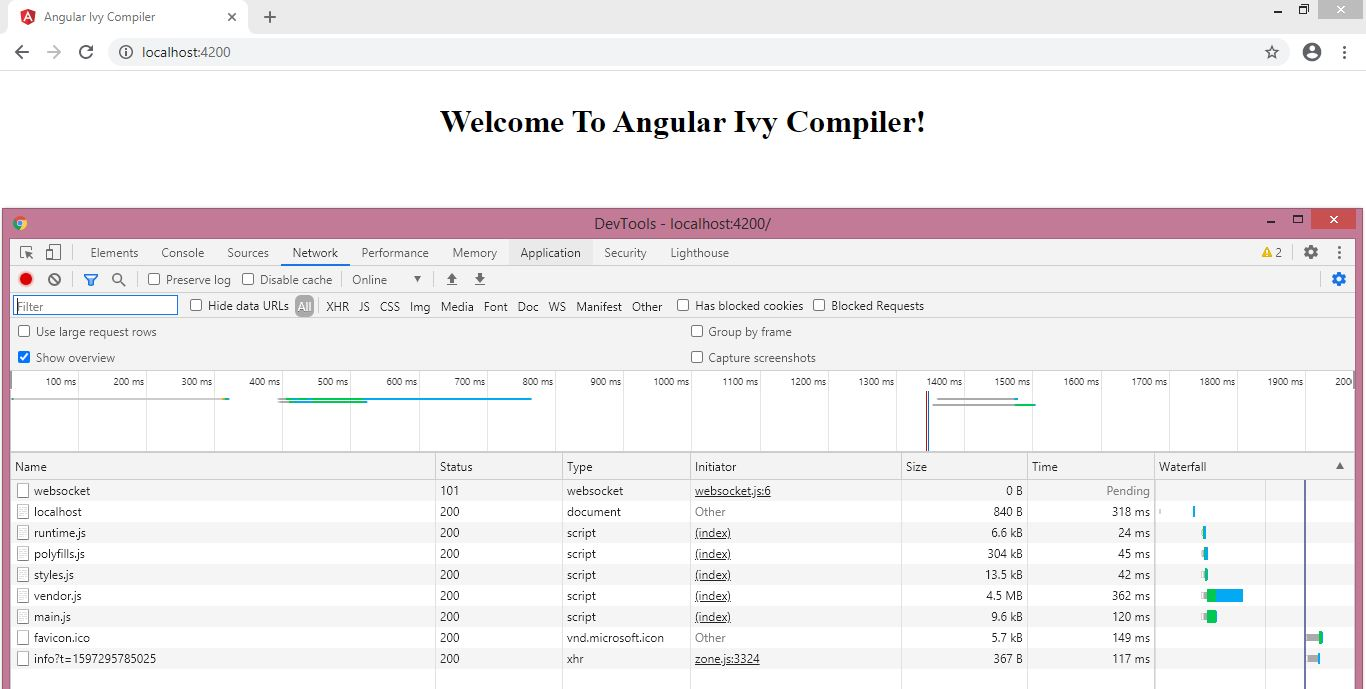
Go to browser: <http://localhost:4200>

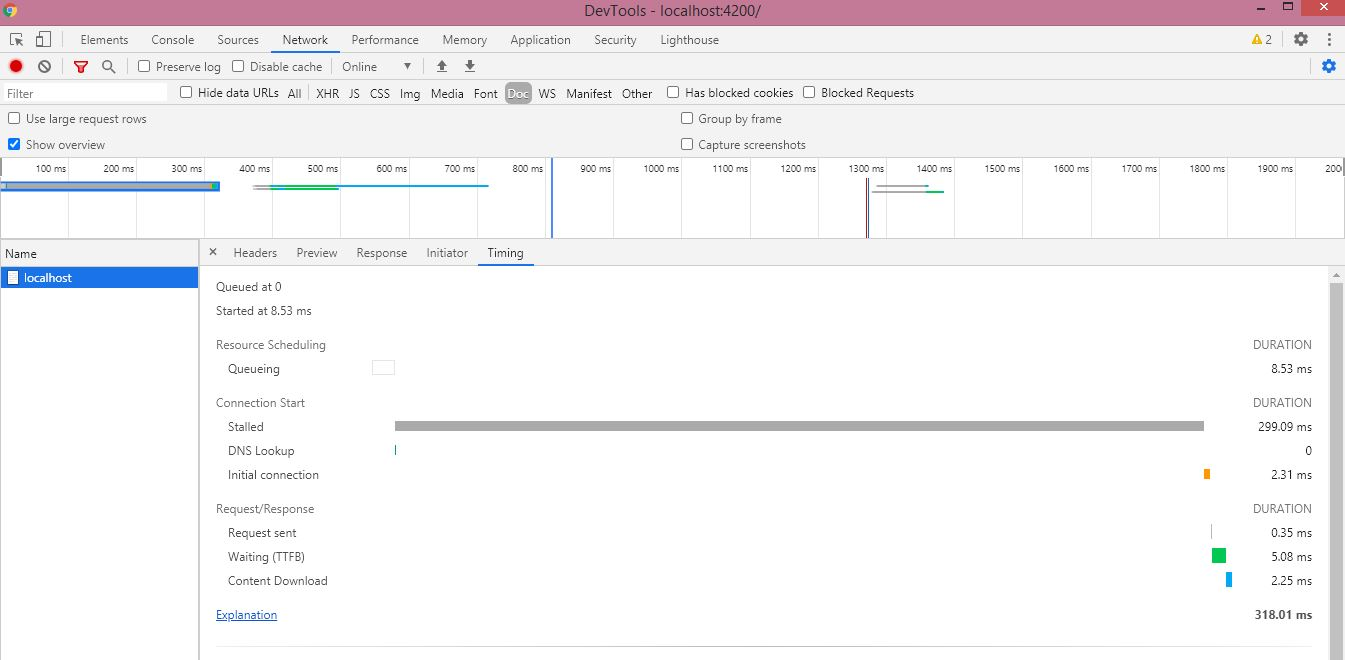


Now see the **Bundle Files**, **Network** and **Timings**:









**Conclusion:**

As we can see, the bundle file size, network connectivity/loading speed and overall timings are much faster than Angular old versions. With Angular 10 and Ivy Compiler we can run any application effectively by keeping efficiency and quality in mind.