

Dependency Injection

2. Injector - The building block of Angular DI

- Used is used to create data, store it, and retrieve that data
 - An example of data can be an instance of one of our Services
- Usually the data is stored in a **Map** like data structure where the keys can be any object
 - Which means a key in the map can be class type, function, string, etc.
- The Injector value for each key starts empty
- When a key is requested the injector:
 - Will create the value if it's the first time and store the value to retrieve later when asked for the key
 - Will retrieve the value if it's already created
- The Injector needs to get a recipe for how to create the value for each key

Injector prototype

```
export declare abstract class Injector {  
    abstract get<T>(token, notFoundValue?: T, flags?: InjectFlags): T;  
  
    static create(options: {  
        providers: StaticProvider[];  
        parent?: Injector;  
        name?: string;  
    }): Injector;  
}
```

Exercise

Supply an Injector with a value

Exercise Injector

- Our goal in this coding exercise is to provide an Injector with the string key: **Hello**
- For the key **Hello** we will supply the value **World**

Summary

- The Injector is the building block of angular's dependency injection mechanism
- We cannot understand how the DI in angular is working before we understand the key ingredient the injector

Thank You

Next Lesson: 3. Injector Trees