

A collection of various blue geometric shapes including triangles, squares, and circles, some containing icons like gears and a lightbulb, arranged in a loose cluster on the left side of the slide.

Angular 2

Component basics
and templates

Angular module

Angular apps are modular and Angular has its own modularity system called Angular modules or **NgModules**.

Every Angular app has at least one module, the root module, conventionally named **AppModule**.

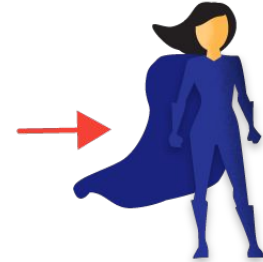
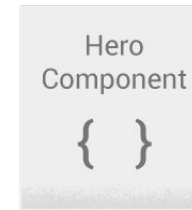
An Angular module, whether a root or feature, is a class with an **@NgModule** decorator.

```
import { NgModule }    from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
@NgModule({
  imports:    [ BrowserModule ],
  declarations: [ AppComponent ],
  exports:    [ AppComponent ],
  bootstrap:  [ AppComponent ]
})
export class AppModule { }
```

app/app.module.ts

Component

app/hero-list.component.ts



```
export class HeroListComponent implements OnInit {  
  heroes: Hero[];  
  selectedHero: Hero;  
  
  constructor(private service: HeroService) { }  
  
  ngOnInit() {  
    this.heroes = this.service.getHeroes();  
  }  
  
  selectHero(hero: Hero) { this.selectedHero = hero; }  
}
```

Template

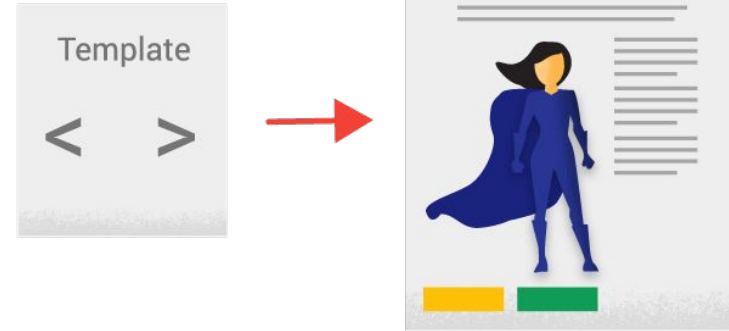
app/hero-list.component.
html

```
<h2>Hero List</h2>
```

```
<p><i>Pick a hero from the list</i></p>
```

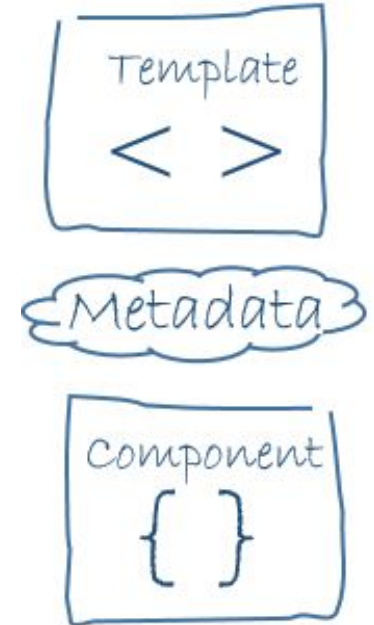
```
<div *ngFor="let hero of heroes" (click)="selectHero(hero)">  
  {{hero.name}}  
</div>
```

```
<p>Selected hero: "{{selectedHero.name}}"</p>
```



@Component decorator

```
@Component({  
  selector: 'hero-list',  
  templateUrl: 'app/hero-list.component.html'  
})  
export class HeroListComponent { ... }
```



- **selector** - a css selector that tells Angular to create and insert an instance of this component where it finds a `<hero-list>` tag in *parent* HTML:
`<hero-list></hero-list>`
Angular inserts an instance of the `HeroListComponent` view between tags.
- **templateUrl** - the address of this component's template

Example: component

```
import {Component} from '@angular/core';

@Component({
  selector: 'my-app',
  template: `
    <h1>{{title}}</h1>
    <h2>My favorite hero is: {{myHero}}</h2>
  `
})
export class AppComponent {
  title = 'Tour of Heroes';
  myHero = 'Windstorm';
}
```

index.html

```
:
<body>
  <my-app>Loading...</my-app>
</body>
```

Example: create class for data and fill it

hero.ts:

```
export class Hero {  
  constructor(public id:number, public name:string) { }  
}
```

app.component.ts:

```
@Component({ selector: 'my-app',  
  templateUrl: 'app.component.html' })  
export class AppComponent {  
  title = 'Tour of Heroes';  
  heroes = [  
    new Hero(1, 'Windstorm'), new Hero(13, 'Bombasto'),  
    new Hero(15, 'Magenta'), new Hero(20, 'Tornado')  
  ];  
  myHero = this.heroes[0];  
}
```

Example: show the list of heroes

Template **app.component.html**:

```
<h1>{{title}}</h1>
<h2>My favorite hero is: {{myHero.name}}</h2>
<p>Heroes:</p>
<ul>
  <li *ngFor="let hero of heroes">
    {{ hero.name }}
  </li>
</ul>
<p *ngIf="heroes.length > 3">There are many heroes!</p>
```


Example: work with the events

```
@Component({
  selector: 'click-me',
  template: `
    <button (click)="onClickMe()">Click me!</button>
    {{clickMessage}}`
})
export class ClickMeComponent {
  clickMessage = '';
  onClickMe() {
    this.clickMessage = 'You are my hero!';
  }
}
```

Example: adding hero form

```
@Component({
  selector: 'add-hero',
  template: `
    <input #newHero (keyup.enter)="addHero(newHero.value)"
      (blur)="addHero(newHero.value); newHero.value=" ">
    <button (click)=addHero(newHero.value)>Add</button>

    <ul> <li *ngFor="let hero of heroes">{{hero}}</li> </ul>
  `})
export class AddHeroComponent {
  heroes=['Windstorm', 'Bombasto', 'Magneta', 'Tornado'];

  addHero(newHero:string) {
    if (newHero) {
      this.heroes.push(newHero);
    }
  }
}
```

Example: execute

main.ts

```
import {platformBrowserDynamic} from '@angular/platform-browser-dynamic'  
import {AppModule} from './app.component'
```

```
const platform = platformBrowserDynamic();  
platform.bootstrapModule(AppModule);
```

package.json

scripts

start: concurrently start TSC transpiler and server

necessary libraries: name, version

dependencies (SystemJS is used as module system)

devDependencies

tsconfig.json

TypeScript configuration

npm start

A horizontal arrangement of various geometric shapes and icons in shades of blue and white. It includes circles, triangles, squares, and icons representing gears, a lightbulb, and a globe.

**Thank you
and have a great Angular 2
experience!**



LXFT
LISTED
NYSE

