ViewChild & ViewChildren

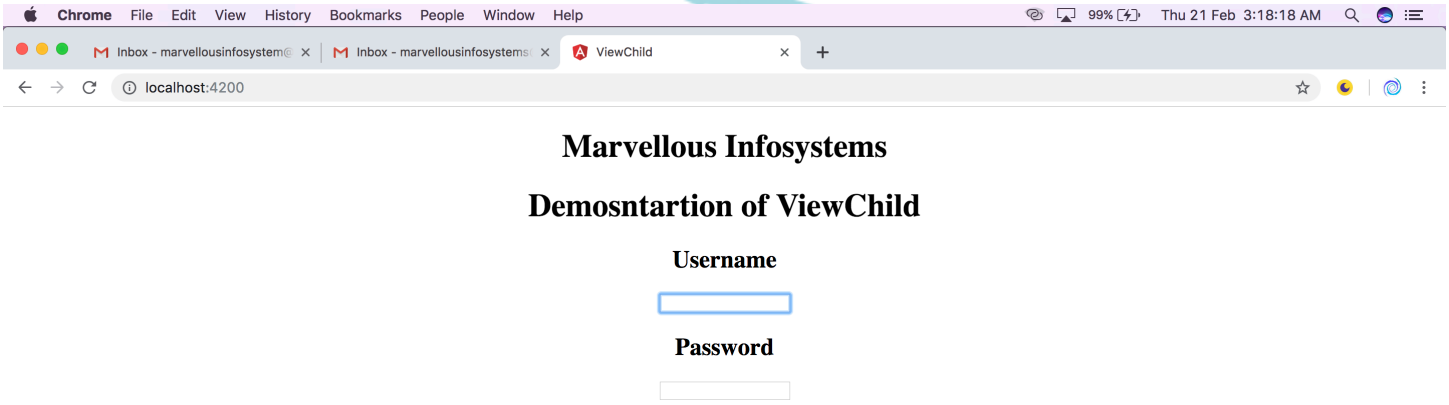
Most of the times when we need to change the behavior or appearance of an element in our template from our component class (i.e. from TS file).
we require a reference variable of that element.

For this, in View (HTML) we can use template reference variable,

but we get this variable into our component class using @ViewChild or @ViewChildren.

Consider our Angular application MarvellousViewChild to understand the concept of ViewChild.

In this application we set focus to textfield using **ViewChild**.



Steps to create the application :

Step 1:
Create a new component called "Marvellous" using the following command,

ng generate component Marvellous

Step 2 :
After that, write the following code in "Marvellous.component.html" file.

```
<h2>Username</h2>
<input #myref type = "text">

<h2>Password</h2>
<input type = "text">
```

Step 3 :
Now, in our component class, to get the reference of this button, we use @ViewChild decorator function.

```
import { Component, OnInit } from '@angular/core';
import { ViewChild, ElementRef, ViewChildren, QueryList } from '@angular/core';

@Component({
  selector: 'app-marvellous',
  templateUrl: './marvellous.component.html'
```

```
})
export class MarvellousComponent
{
  @ViewChild("myref") myValue:ElementRef;

  ngAfterViewInit()
  {
    console.log(this.myValue);
    this.myValue.nativeElement.focus()
  }
}
```

Now, if we want to access any property of child component into parent then we can use **ViewChildren**.

Consider our Angular application MarvellousViewChildren to understand the concept of ViewChildren.

In this application we have one child component which displays date and time.
From Parent component that child component gets rendered.

Using ViewChildren we access property of child components from parent components and write the contents into log.

Steps to create the application :

Step 1:
Create a new component called "Child" using the following command,

ng generate component Child

Step 2:
Create a new component called "Parent" using the following command,

ng generate component Parent

Step 3 :
Inside .html file of child component add below statement to display clock
{{today|date:'mediumTime'}}

Step 4:
Inside .ts file of child component add below code as

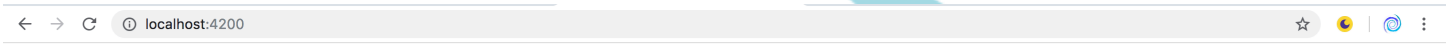
```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-child',
  templateUrl: './child.component.html'
})
export class ChildComponent
{
  today:Date=new Date();
}
```

Step 5:
Inside Parent component render child component twice with one button.

```
<div class="container">
    <div class="row mt-2 mb-2 text-center">
      <div class="col-lg-12 col-md-12 col-sm-12">
       <app-child></app-child>
```

```
    </div>
   </div>
   <div class="row mt-2 mb-2 text-center">
     <div class="col-lg-12 col-md-12 col-sm-12">
      <app-child></app-child>
     </div>
   </div>
   <div class="row mt-2 mb-2 text-center">
     <div class="col-lg-12 col-md-12 col-sm-12 col-12">
      <button type="button" class="btn btn-primary" #myButton>My Button</button><br>
    </div>
    </div>
  </div>
```



Marvellous Infosystems

Demosntartion of ViewChildren

3:20:19 AM
3:20:19 AM
My Button

Step 6 :
Inside .ts file of parent component access child property using ViewChildren as

```
import { ChildComponent } from './../child/child.component';
import { Component, OnInit, ViewChild, ElementRef, ViewChildren, QueryList } from '@angular/
core';
import {Directive, Input} from '@angular/core';
@Component({
  selector: 'app-parent',
  templateUrl: './parent.component.html'
})

export class ParentComponent
{
  @ViewChildren(ChildComponent) myValue:QueryList<ChildComponent>;

     ngAfterViewInit()
     {
       console.log(this.myValue.toArray());
     }
}
```

After executing the application check log.