Angular frontend fejlesztés (TCS- Tata Consultancy Services)
1. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 20 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 1 pont

Az alábbi állítások közül melyek igazak a komponensekre?

- [x] a komponensre (@Component) mondhatjuk, hogy tulajdonképpen direktíva, amelyhez tartozik template
- [ ] class selectorral ugyanazon a DOM elemen több komponens is használható
- [x] egy DOM elemen több direktívát is használhatunk
- [x] a komponenshez kötelezően tartoznia kell templatenek, amelyet a template vagy templateUrl-ben meg kell adni
- [x] a direktíva funkciója, hogy létező DOM elemek viselkedését kiegészítse/módosítsa
- [x] komponenseket használva a kódunk kisebb egységekre bontható

Magyarázat a megoldáshoz

2. feladat 0 / 8 pont

Az alábbiak közül melyek igazak az Angularra?

- [ ] legújabb verziója a 9, mely az Ivy nevet kapta
- [ ] az Angular (2+) az AngularJs továbbfejlesztett változata, amelyet nem kellett újraírni, csak átnevezték, mert Typescriptben nem elérhető a $scope változó
- [ ] az Angularban is megtalálható az AngularJs-ből ismert direktíva, pipe, controller, service, csak külön modulokba vannak szervezve
- [ ] az Angular csak Typescripttel használható, ezért nem kell szenvedni a Javascript megtanulásával, amely fejlesztését már nem is erőltetik
- [x] a beépített pipe-ok mellett lehetőség van saját, custom pipe-ok létrehozására is egyedi megjelenítésekhez
- [ ] a CLI – Common Language Interface a vele generált komponenseket automatikusan át tudja fordítani javascriptre, ezért futtatható a böngészőben
- [ ] egyszerre megadható külön html fájlban és inline template-ben egy komponenshez tartozó view kódja, amelyet ilyen esetben összefésül a fordító
- [x] a direktíváknak 3 típusa van: Component, Structural Directive, Attribute Directive
- [x] Angularban mind a 4-féle typescript decoratorra találhatunk példát
- [ ] az *NgIf egy structural directive, ami nem támogatja az else ágat, szükség esetén *ngElse directive-t lehet használni
- [ ] az ngShow és ngHide direktívák deprecatedek, csak az AngularJs támogatása miatt maradtak a kódban, helyettük a hidden attribútum használata javasolt
- [x] egy component működéséhez szükség van minimum a selector és a template (vagy templateUrl) megadására
- [x] az ng g c <name> -S paranccsal egy új komponenst tudunk generálni parancssorban, melyhez nem jön létre automatikusan a <name>.spec.ts fájl
- [x] az Angular Universal server-side rendering modul Angular alkalmazásokhoz
- [x] a hidden attribútum használatakor az adott elem a DOM-ban elérhető, csak a display property értéke lesz 'none', *NgIf esetén viszont az egész elem ki van törölve a DOM-ból

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
2. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 20 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 1 pont

Az alábbiak közül melyik direktíva selector hibás?

- ○
```
@Directive({
    selector: '[digitOnly]'
})
```
- ○
```
@Directive({
    selector: '.myClass'
})
```
- ○
```
@Directive({
    selector: 'div'
})
```
- ○
```
@Directive({
    selector: 'div, [digitOnly]'
})
```
- ◉
```
@Directive({
    selector: 'div+.myClass'
})
```

Magyarázat a megoldáshoz

2. feladat 0 / 10 pont

Mely állítások igazak a pipe-pal kapcsolatban?

- ☐ a name és a pure paramétert kötelező megadnunk
- ☑ a pipe feladata, hogy a kapott inputot a kívánt outputra alakítsa át
- ☑ a pipe-ok chainelhetők
- ☑ custom pipe esetében implementálni kell a PipeTransform interface-t, aminek van egy transform metódusa
- ☑ a custom pipe tulajdonképpen egy typescipt osztály, ami a @Pipe decoratorral van ellátva
- ☑ az async pipe lehetőséget ad, hogy a template-ünkben streamet használhassunk
- ☑ az async pipe automatikusan kezeli a fel/leiratkozást az observable-ről/streamről
- ☑ a pipe fogadhat opcionális paramétert (| date:'dd/MM/yyyy')
- ☐ a pure pipe-ról elmondható, hogy gyakrabban hívódik meg, mint az impure pipe

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
3. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 30 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 10 pont

Mi íródik ki a konzolba?

```
class A {
    public log() {
        console.log('log: Class A');
    }
}

class B extends A {
    public log() {
        console.log('log: Class B');
    }
}

const a: A = new B();
const b: B = new B();

a.log();
b.log();
```

- log: Class A
  log: Class B
- log: Class B
  log: Class B
- log: Class A
  log: Class A
- hibás a kód

Magyarázat a megoldáshoz

2. feladat 0 / 10 pont

Az alábbi egyszerű pipe-ot hoztuk létre gyakorlásként. Tegyük fel, hogy a mai dátum 2020. május 10. Válaszd ki, hogy a felsorolt lehetőségeket az app.component 9.sorába helyettesítve melyik esetben jelenik meg a Formatted date is: 2020-05-10 kiírás!

my-pipe.pipe.ts:

```
import { Pipe, PipeTransform } from '@angular/core';
@Pipe({
  name: 'myPipe'
})
export class MyPipe implements PipeTransform {
  transform(value: Date): string {
    return ${value.getFullYear()} - ${value.getMonth()} - ${value.getDat
  }
}
```

app.component.html:

```
<div>
    <h1>Formatted date is: {{currentDate | myPipe}}</h1>
</div>
```

app.component.ts:

```
import { Component } from '@angular/core';
@Component({
  selector: 'my-app',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  currentDate: Date;
  ngOnInit() { }
}
```

- currentDate: Date;
- currentDate = Date.now();
- egyik sem
- currentDate = new Date();

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
4. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 40 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 20 pont

A felsorolt kódrészletek közül melyek fordulnak az alábbi javascript kóddá?

```
"use strict";
class MyClass {
    constructor() {
        this.myProperty = 1;
    }
    myMethod() {
        console.log(this.myProperty);
    }
}
```

- [ ]
```
class MyClass {
    var myProperty: number;

    constructor() {
        this.myProperty = 1;
    }

    myMethod() {
        console.log(this.myProperty)
    }
}
```

- [x]
```
class MyClass {
    private myProperty: number;

    constructor() {
        this.myProperty = 1;
    }

    myMethod() {
        console.log(this.myProperty)
    }
}
```

- [x]
```
class MyClass {
    public myProperty: number;

    constructor() {
        this.myProperty = 1;
    }

    myMethod() {
        console.log(this.myProperty)
    }
}
```

- [x]
```
class MyClass {
    readonly myProperty: number;

    constructor() {
        this.myProperty = 1;
    }

    myMethod() {
        console.log(this.myProperty)
    }
}
```

- [x]
```
class MyClass {
    const myProperty: number;

    constructor() {
        this.myProperty = 1;
    }

    myMethod() {
        console.log(this.myProperty)
    }
}
```

Magyarázat a megoldáshoz

2. feladat 0 / 10 pont

Az alábbi változó deklarációk közül melyek hibásak typescriptben?

- [ ]
```
let variable = 100;
```

- [ ]
```
let variable: number = 100;
```

- [ ]
```
let variable;
```

- [x]
```
let number variable = 100;
```

- [ ]
```
let variable: number;
```

- [x]
```
number variable;
```

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
5. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 40 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 20 pont

Válaszd ki, hogy melyik javascript kódrészletté fordul a következő typescript kód!

```
interface ITeszt {
    setValue(value1: number): void;
}

class MyClass implements ITeszt {
    setValue(value1: number): void {
        console.log(value1)
    }

    myMethod(a: string): string;
    myMethod(a: number): string;
    myMethod(a: number, b: string): string;
    myMethod(a: any, b?: string) {
        return 'teszt';
    }
}
```

- (●)
```
"use strict";
class MyClass {
    setValue(value1) {
        console.log(value1);
    }
    myMethod(a, b) {
        return 'test';
    }
}
```

- ( )
```
"use strict";
{
    setValue(value1, number);
    void ;
    { }
}
class MyClass {
    setValue(value1) {
        console.log(value1);
    }
    myMethod(a, b) {
        return 'test';
    }
}
```

- ( )
```
"use strict";
{
    setValue(value1, number);
    void class MyClass {
        setValue(value1) {
            console.log(value1);
        }
        myMethod(a, b) {
            return 'test';
        }
    }
};
```

- ( )
```
"use strict";
class MyClass {
        setValue(value1) {
            console.log(value1);
        }
        myMethod(a) { }
        ;
        myMethod(a) { }
        ;
        myMethod(a, b) { }
        ;
        myMethod(a, b) {
            return 'test';
        }
}
```

Magyarázat a megoldáshoz

2. feladat 0 / 20 pont

A következő lehetőségek közül melyek adnak a 15. sorral megegyező eredményt?

```
interface IEmployeeResult {
    employee?: {
        id?: any
    }
}

function getResponse(): IEmployeeResult {
    //... returns IEmployeeResult obj
}

const response = getResponse();

const result = response ? (response.employee ? response.employee.id : ur

console.log(result);

console.log(((response || {}).employee || {}).id);

console.log(response?.employee?.id);

console.log((response && response.employee && response.employee.id) ? re

console.log((response || response.employee || respose.employee.id) ? res

const response: IEmployeeResult = {
    employee: {
        id: {}
    }
}
```

- [x] 17-es sor
- [x] 19-es sor
- [x] 21-es sor
- [ ] 23-as sor

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
6. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 50 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 20 pont

Melyik állítás igaz?

app.component.ts:

```
import { Component, VERSION } from "@angular/core";

@Component({
  selector: "my-app",
  templateUrl: "./app.component.html",
  styleUrls: ["./app.component.css"]
})
export class AppComponent {
  currentValue: number;

  getValue1() {
    return 0.1;
  }
  getValue2() {
    return 0.2;
  }

  isValid() {
    const threshold = 0.3;
    this.currentValue = this.getValue1() + this.getValue2();
    return this.currentValue >= threshold;
  }
}
```

app.components.html:

```
<div>
    {{currentValue | number:'1.1-17'}}
    <h1 [hidden]="!isValid()">VALID</h1>
</div>
```

- megjelenik a 0.3-as érték és a VALID szöveg nem látható
- megjelenik a 0.3-as érték és a VALID szöveg is látható
- 0.30000000000000... íródik ki és nem jelenik meg a VALID szöveg
- 0.30000000000000004 íródik ki és megjelenik a VALID szöveg

Magyarázat a megoldáshoz

2. feladat 0 / 20 pont

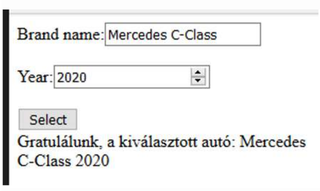
Melyik design patternt próbáljuk használni az alábbi kódban? Az app.componentben 2 komponenst használunk. Az egyik egy autóválasztó, ahol az autó nevét és évjáratát kell megadni. Ezt emittáljuk a tartalmazó komponensnek, ami az adatokat továbbküldi egy konfirmációs komponensnek, amely megjeleníti ezeket az adatokat. Az app.componentnek nagy szerepe van a kommunikációban, hiszen kapcsolatban van a másik 2 komponenssel, amelyek nem közvetlenül egymással kommunikálnak.

app.component.html:

```
<app-car-selector (select)="carSelectionHandler($event)"></app-car-selec
<br/>
<app-confirmation [selectedCarDetails]="car"></app-confirmation>
```

app.component.ts:

```
import { Component, VERSION } from "@angular/core";
import { Car } from "./car";

@Component({
  selector: "my-app",
  templateUrl: "./app.component.html",
  styleUrls: ["./app.component.css"]
})
export class AppComponent {
  car: Car;

  carSelectionHandler(event: Car) {
    this.car = event;
  }
}
```



- Iterator
- Mediator
- Observer
- Semmilyen, nem jól sikerült a példa

Magyarázat a megoldáshoz

Vissza a kategóriáimhoz

Angular frontend fejlesztés (TCS- Tata Consultancy Services)
7. forduló

Ismertető a feladathoz

Válaszd ki a helyes választ/válaszokat és figyelj oda, hogy az összeset kiválaszd, mert csak akkor kapsz pontot.

Tekintettel arra, hogy egy választ sem rögzítettél az alábbi feladatlapon, ebben a fordulóban a kitöltésére rendelkezésre álló idő teljes egésze, azaz 50 perc került rögzítésre mint megoldáshoz felhasznált idő.

1. feladat 0 / 20 pont

Válaszd ki, hogy milyen sorrendben íródnak ki a konzolra az angular hookok az alábbi példában!

2 egyszerű komponenst készítünk el. Az első (static nevű) egyszerű üdvözlő komponens lesz, ami most egy default szöveget ír ki.

static.component.ts:

```
import { Component, OnInit, OnChanges, AfterContentInit, OnDestroy } frc

@Component({
  selector: "app-static",
  templateUrl: "./static.component.html",
  styleUrls: ["./static.component.css"]
})
export class StaticComponent implements OnInit, OnChanges, AfterContentI
  constructor() {
    console.log("STATIC ctor");
    setTimeout(() => this.ngOnDestroy(), 10000); //feltételezzük, hogy m
  }

  ngOnChanges() {
    console.log("STATIC ngOnChanges");
  }

  ngOnInit() {
    console.log("STATIC ngOnInit");
  }

  ngAfterContentInit() {
    console.log("STATIC ngAfterContentInit");
  }

  ngOnDestroy() {
    console.log("STATIC ngOnDestroy");
  }
}
```

static.component.html:

```
<p>Üdvözöljük</p>
```

app.component.html:

```
<app-static></app-static>
```

- ctor – ngOnChanges – ngOnInit – ngOnDestroy
- ngOnChanges – ctor - ngOnInit – ngOnDestroy
- ngOnInit – ngOnChanges - ngAfterContentInit – ngOnDestroy
- **ctor – ngOnInit – ngAfterContentInit – ngOnDestroy**
- ngOnInit – ngAfterContentInit – ngOnDestroy

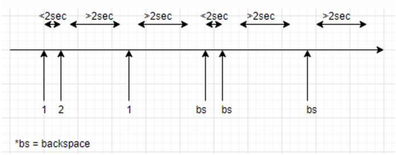
Magyarázat a megoldáshoz

2. feladat 0 / 20 pont

Az alábbiakban egy egyszerű keresés funkciót próbálunk megvalósítani. Adott egy filter operátor, illetve az elvárás az, hogy kattintás nélkül induljon el a keresés és az elgépelések miatt legyen egy 2 másodperces határ, amin belül nem küldünk ki hívást. Alább a kódrészletünk:

```
const { fromEvent } = Rx;
const { map, filter, tap, debounceTime,
        distinctUntilChanged, pluck } = RxOperators;

const input = document.createElement('input');

input.setAttribute('placeholder', 'Type something');

// `output` represents the right hand pane.
// You can prepend/append elements to it.
output.prepend(input);

input.focus();

fromEvent(input, 'input').pipe(
  pluck('target', 'value'),
  filter(id => id >= 0 && id <= 100),
  distinctUntilChanged(),
  debounceTime(2000)
)
```

A következő billentyűleütésekkel szimuláljuk a tesztünket:



A 0,5s és 3s az időtartamot jelentené, h 3sec, az 1,2 és bs(backspace) pedig a billentyú leütések.

tehát a textboxban kb. ez van 2 másodpercenként:

üres

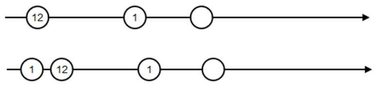
12

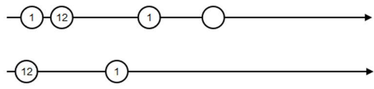
121

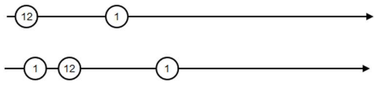
1

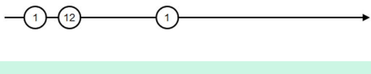
üres

Ez alapján melyik diagram tartozik a példához?

- **  **

- 

- 

- 

Magyarázat a megoldáshoz