QNH - WORKSHOPS ANGULAR, OKTOBER 2017

01 – EENVOUDIG BEGINNEN

a) Het algemene Github-adres met voorbeeldcode is github.com/PeterKassenaar/voorbeeldenAngular2

b) Maak/start een eigen Hello World-app. Dit kan op verschillende manieren:
   - Gebruik als basis `100-helloworld`.
   - OF: ga zelf naar `cli.angular.io`, installeer deze tool en maak een nieuw project en start deze. Lees hiervoor zelf de online instructies (zie ook 1d.).
   - Codevoorbeeld: `100-helloworld`.

c) Probeer eens of je een nieuwe component kunt maken en deze bootstrappen. De nieuwe component komt dan *in plaats van* `<hello-world>`, nog niet erbij. Volg deze stappen:
   - *Handmatig?* - maak een nieuw bestand, bijvoorbeeld \app\nieuwe.component.ts.
     - Importeer de juiste afhankelijkheden.
     - Schrijf een selector en een HTML-template.
   - *CLI?* - gebruik de opdracht `ng generate component <component-naam>`.
   - Pas de bootstrapper aan, zodat daarin nieuwe component wordt geïmporteerd en gestart.
   - Pas `index.html` aan, zodat de juiste selector wordt gebruikt.
   - Draai `npm start` om je nieuwe component te testen.

d) Installeer Angular-CLI en maak hiermee een basisproject.
   - De aanwijzingen hiervoor vind je op https://cli.angular.io/ .
   - Maak een nieuw project met `ng new <projectnaam>`.
   - Gebruik de opdrachten `ng new`, `ng serve` en `ng generate`. Zoek zelf op hoe dit werkt.

02 – DATABINDING

a) Breid je app uit oefening 1 (`Hello World`) uit met een field/property. Bindt deze property in het template dat gebruikt wordt.
   - Doe dit zowel impliciet (met rechtstreekse initialisatie van variabelen) als expliciet (met aparte declaratie en initialisatie in de `constructor`).
   - Gebruik daarna `ngOnInit()` voor initialisatie van je variabele.
   - Lees de werking en de volgorde van Angular lifecycle hooks op https://angular.io/guide/lifecycle-hooks.
   - Codevoorbeeld: `101-databinding`.

b) Maak een array van properties. Bind ze in het template met de directive `*ngFor`.
   - Geef via TypeScript expliciet op dat de property uit een array van strings of een array van objecten moet bestaan (maak zelf deze keuze).
   - Codevoorbeeld: `101-databinding`.

03 – EVENT BINDING

a) Voeg een element met event-binding toe aan je applicatie. Bijvoorbeeld een `(click)` op een knop afvangen. Roep een event handler in de component aan als de event optreedt. Laat deze bijvoorbeeld een `alert()` tonen of een `console.log()`-melding.
   - Codevoorbeeld: `102-eventbinding`.

b) Maak een eenvoudige client-sided CRUD-applicatie: users kunnen elementen toevoegen aan een array (namen, producten, enzovoort) en verwijderen uit de array. Gebruik voor verwijderen de JavaScript-functie `ArrayName.splice(…)`.
   - Codevoorbeeld: `102-eventbinding\...\app-02-complete.component.ts`.

04 - SERVICES

a) Als het goed is heb je inmiddels enkele componenten met hierin diverse data. Verplaats deze data vanuit de component naar een service.
   - Injecteer de service in de constructor van de component waarin je de data wilt gebruiken.
   - Denk aan de property `[providers]` in de `@ngModule`-annotatie.
   - Codevoorbeeld : `200-services-static`.

b) Werken met async-services: maak zelf een .json-bestand met data en laadt dit in je applicatie. Denk onder meer aan:
   - het injecteren van `HttpModule` in de module;
   - het aanpassen van de component waarin de service wordt geconsumeerd: `private http:Http` injecteren in de constructor en de observable-notatie met `.subscribe()` gebruiken.
   - Codevoorbeelden: `0201-services-http` en `202-services-rxjs`.

c) Werken met Live API's: gebruik een API naar keuze om live data op te halen. Maak een nieuwe applicatie en gebruik hierin deze data.
   - API's zijn bijvoorbeeld beschikbaar op `openweathermap.org/API` of `www.omdbapi.com`.
   - Zie ook de site https://github.com/toddmotto/public-apis en het bestand `JavaScript APIs.txt` voor meer API-endpoints.
   - Codevoorbeeld: `210-services-live`.