QNH WORKSHOPS – OKTOBER 2017, ECMASCRIPT 2015, TYPESCRIPT EN NODEJS

01 – KORTE KENNISMAKING - COMPATIBLITY TABLE

a) Bekijk de ECMAScript 2015-compatilibity table op https://kangax.github.io/compat-table/es6/. Leer deze lezen en begrijpen voor de verschillende browsers. Bekijk:
   - De nieuwe keywords aan de linkerkant
   - De generatie Compilers/transpilers en polyfilss (Traceur, Babel, TypeScript).
   - De generatie browsers. Hoe scoort jouw browser op dit moment?

02 – LET EN CONST

a) Schrijf een script waarin een `var` wordt gebruikt in een lus. Laat de lus doorlopen en check of de variabele na afloop van de lus nog steeds bestaat.
   - Vervang de var door een `let` en controleer of de variabele nog steeds bestaat na afloop van de lus. Hoe verklaar je het verschil?
   - Probeer er nu een `const` van te maken. Lukt dit in een lus? Zo nee, waarom niet?

03 - ARROW FUNCTIONS

a) Schrijf een functiedeclaratie op de 'ouderwetse' manier (ES5), met parameters, een functiebody en een return-resultaat. Maak bijvoorbeeld een function `optellen(a, b)` die de som van `a` en `b` retourneert.
   - Herschrijf deze functie zodanig dat nu gebruik wordt gemaakt van een arrow-function.
   - Doe hetzelfde voor een andere functie die je schrijft.
   - Compileer en test de functie in de browser.
   - Schrijf de functie op twee manieren: in één regel (geen `{ … }` nodig) en over meerdere regels. Hierbij is wel `{ … }` nodig, en kan bijvoorbeeld binnen de functiebody het resultaat ook naar de console worden gelogd. Vergeet nu niet om expliciet het statement `return` te gebruiken.

04 - CLASSES

a) Gebruik eventueel het bestand `0303-constructor.js` uit de downloadbestanden als voorbeeld.
   - Maak zelf een klasse `Person`, schrijf hierin een methode `getFullName()` die de volledige naam van de persoon retourneert.
   - Breid de klasse uit met velden voor adres, postcode en woonplaats. Deze worden bij initialisatie van de instantie doorgegeven.
   - Schrijf een `get-` en een `set`-methode voor deze eigenschappen, zodat ze vanuit de instantie gewijzigd kunnen worden.
   - De notatie wordt dan bijvoorbeeld `persoon.setAdres('Dorpsstraat 9');`. Gebruik daarna een `console.log()` om te checken of het nieuwe adres goed is geregistreerd.

WORKSHOPS TYPESCRIPT

05 – INSTALLATIE EN GEBRUIK

a) Installatie en gebruik
   - Installeer Node.js (indien nodig)
   - Installeer TypeScript globaal. Aanwijzingen hiervoor zijn te vinden op http://www.typescriptlang.org/#download-links.

- Schrijf een kort Hello World-programma en compileer dit (van `helloworld.ts` naar `helloworld.js`)
- Test de uitvoer in de browser.
- Een voorbeeld is beschikbaar in `workshops/ts/0401-hello-world.ts`.

06 - WERKEN MET BASIC TYPES

a) Werken met de basis gegevenstypen.
   - Begin een TypeScript-bestand en maak in een pagina een variabele van het gegevenstype boolean. Sla hierin een waarde `true` of `false` op.
   - Probeer op een nieuwe regel hier een andere waarde in op te slaan, bijvoorbeeld een getal of een tekst. Welke foutmelding geeft de TypeScript-compiler?
   - Verwijder het gegenereerde JavaScript-bestand.
   - Wijzig het gegevenstype in any. Compileert de code nu correct?
   - Codevoorbeeld: `0501-types.ts`.

b) Maak op een nieuwe regel een array van strings. Gebruik de correcte TypeScript-typering. Loop met een lus `.forEach()` over alle items in de array en toon ze in de console.

c) Doe hetzelfde, maar dan met een ander arraynotatie, bijvoorbeeld number, of een array van any gegevens.

07 - WERKEN MET INTERFACES

a) Werken met interfaces.
   - Maak een interface Auto met de gewenste kenmerken.
   - Definieer twee of drie `auto`'s op basis van de interface.
   - Schrijf een functie die een Auto-object teruggeeft op basis van parameters. Zorg voor de correct typering van het returntype in TypeScript. Maak nu opnieuw twee of drie autos.

b) Sla de auto's op in een array met het type van de interface. De notatie wordt dus iets als `Auto[]`, of `Array<Auto>`. Gebruik daarna `.forEach()` om elke auto in de console of in de webpagina te tonen.

c) Bekijk de door TypeScript gegenereerde JavaScript-code. Kun je de eigenschappen van `Auto` terugvinden?
   - Codevoorbeeld: `0502-interface.ts` (hier wordt een interface `Person` gebruikt, wijzig dit dus zelf in `Auto`, of -liever nog – schrijf helemaal *from scratch*).

08 – WERKEN MET CLASSES

a) Werken met classes.
   - Maak nu ook een `class Auto` en geef deze dezelfde eigenschappen.
   - Zorg ervoor dat de klasse dezelfde eigenschappen heeft als de interface. Laat deze als eerste definiëren door de shorthand notatie `constructor (public <propertyName1>, public <propertyName2>, …)` te gebruiken.
   - Verwijder daarna de keywords `public` uit de constructor en schrijf interne `private` variabelen en maak deze bereikbaar via de keywords `get` en `set`.
   - Zie bijvoorbeeld `0504-private.ts` uit de downloadbestanden voor een voorbeeld.