Module 04 Text Adventure

READER v1.1

AUTEUR:
Team Software Developer

SOFTWARE DEVELOPER

Alfa-college

Inleiding

In deze reader staat beschreven wat er tijdens deze module van jou verwacht wordt en kun je belangrijke informatie over deze module terugvinden.

In deze module leer je hoe je een text adventure game ontwikkelt met C#. Je leert object-georiënteerd programmeren (OOP) en werkt met belangrijke programmeerconcepten zoals classes, methods, properties en collections.

Software Engineering betekent: goede code schrijven.

Goede code is o.a.:
- Onderhoudbaar (je kunt het later makkelijk aanpassen)
- Gedocumenteerd (er staat uitleg bij)
- In versiebeheer (je gebruikt git)
- Efficiënt (het werkt snel)
- Leesbaar (andere programmeurs begrijpen het)

Uitgebreidere informatie over deze module vind je verderop in deze reader.

Veel succes en plezier tijdens deze module!

- Team Software Developer

Inhoudsopgave

1. Introductie

Welkom bij de opdrachtenreeks **World of Zuul**. In deze lessen ga je stap voor stap leren hoe je met C# een text adventure game bouwt. Het einddoel is een werkende game waarin spelers kamers kunnen verkennen, items kunnen verzamelen en uitdagingen kunnen overwinnen.

1.1 WAT GA JE LEREN?

Object-Georiënteerd Programmeren:

- Hoe je classes ontwerpt en gebruikt (Player, Room, Item, Inventory)
- Hoe je met properties werkt (CurrentRoom, Health, Weight)
- Hoe je methods implementeert (Damage(), Heal(), Put(), Get())
- Hoe je relaties tussen classes maakt (compositie en aggregatie)

Collections en Datastructuren:

- Hoe je een Dictionary gebruikt om items op te slaan
- Verschil tussen List en Dictionary
- Loops door collections

C# Programmeerconcepten:

- Command pattern voor gebruikersinvoer
- Switch statements voor command processing
- Boolean logica voor game states
- Constructor methods
- Error handling

Software Ontwerp:

- UML klassediagrammen maken met yEd
- Code structuur plannen
- Separation of concerns toepassen

Game Development:

- Game loop implementeren
- Win/lose condities maken
- Item systeem bouwen
- Health en inventory management

Onderzoekende houding

In deze module ga je een bestaande basis game uitbreiden. Je moet je de bestaande code-base goed begrijpen voordat je uitbreidingen kunt schrijven. De eerste vervolg stappen zijn ook beschreven in de (bestaande) documentatie. Je zult deze documentatie ook goed moeten bestuderen om verder te kunnen werken aan het eindproduct.

1.2 VOORKENNIS

- Basiskennis van variabelen, if-statements, loops
- Kunnen werken met Visual Studio Code
- Basiskennis van command line (`dotnet run`)

1.3 Waarom leer je dit?

De skills die je in dit project leert gebruik je in de beroepspraktijk:

- Object-Georiënteerd Programmeren: Vrijwel alle moderne software wordt OOP gebouwd
- Collections beheren: Elke app die data verzamelt gebruikt Dictionary of List, als voorbeelden:
- Webshop: producten in winkelwagen
- Schoolsysteem: studenten in een klas
- CRM systeem: klanten in een database
- UML diagrammen: Software architecten en developers gebruiken dit om systemen te ontwerpen
- Code lezen en begrijpen: In je BPV werk je met bestaande code van anderen
- Debugging: Systematisch problemen oplossen is een kernvaardigheid

Competenties: Dit project traint de volgende competenties uit het kwalificatiedossier:

- 1.2 - Software ontwikkelen met objectgeoriënteerde technieken
- 1.3 - Software testen
- 2.1 - Ontwerpen van software (UML)

2. De opdracht

In dit hoofdstuk lees je wat er van jou wordt verwacht tijdens deze module. Als iets niet duidelijk is, kun je dit altijd navragen bij je docent.

2.1 HOE WERKEN DE OPDRACHTEN?

Structuur

- Je werkt 5 weken aan Zuul-opdrachten en UML opdrachten
- Elke week heb je 6 contactmomenten:
  - 4× Instructiemoment (uitleg + begeleide verwerking)
  - 2× Verwerkingsmoment (zelfstandig werken)
- Bij elke opdracht staan exacte specificaties voor class-, method- en property-namen

BELANGRIJK: Gebruik de namen exact zoals aangegeven in de documentatie en opdrachten, anders werkt de code niet goed samen.

Werkwijze

- Je werkt zelfstandig aan de opdrachten
- Kom je er niet uit? Gebruik de hulpdocumenten op Itslearning
- Vraag een medestudent of docent om hulp
- Test je code na elke opdracht met dotnet run

2.2 WAT LEVER JE OP?

Aan het eind van deze module lever je de volgende zaken op.

De game:

- Hele Zuul project map (gezipt)
- Je link naar je GitHub waar je aantoonbaar vanaf het begin in hebt gewerkt.
- Minimaal 6-8 kamers
- Alle opdrachten geïmplementeerd
- Code compileert zonder errors

UML diagrammen

- - `dynamite.png` - Klassediagram van UML Opdracht 1
- - `zuul_final.png` - Klassediagram van jouw Zuul game
- - Beide geëxporteerd uit yEd

README.md

- Beschrijving van je game
- Hoe start je het spel
- Welke speciale features heb je toegevoegd

2.3 REFLECTIE

Zorg ervoor dat je de onderstaande vragen kunt beantwoorden tijdens de reflectieweek:

- Wat heb je geleerd op technisch vlak?
- Wat heb je geleerd op persoonlijk vlak?
- Wat zijn je valkuilen?
- Wat wil je een volgende keer als leerpunt(en) aanpakken?

3. Resultaat

3.1 WAAR LETTEN DE DOCENTEN OP?

Je wordt beoordeeld op 5 criteria. Hieronder de rubric met drie niveaus:

Criterium	A - Basis	B - Goed	C - Excellent
Werkende Code	Code compileert en draait. Basis functionaliteit werkt (navigeren, look, status). Minimaal 6 kamers.	Alle opdrachten 1-11 geïmplementeerd. Game is speelbaar van start tot eind. 8+ kamers. max 10	Game is volledig functioneel en gepolijst. Extra features toegevoegd. Creatieve implementatie. Geen bugs.
Code Kwaliteit	Code is correct ingesprongen. Namen zijn redelijk duidelijk. Weinig duplicatie.	Code is netjes georganiseerd. Logische method namen. Goede separation of concerns. Commentaar waar nodig.	Code is professioneel. Excellente naamgeving. Perfect ingesprongen. Herbruikbare methods. Duidelijke code structuur.
UML Diagrammen	Beide UML opdrachten ingeleverd. Basis classes aanwezig. Relaties getekend.	UML is correct en compleet. Alle classes met properties en methods. Juiste UML notatie. Duidelijk leesbaar.	UML is professioneel. Perfecte notatie. Alle details correct. Visueel uitstekend. Zou gebruikt kunnen worden als documentatie.
Versiebeheer	Er is een git en github aanwezig met een aantal commits	Redelijk omschreven commits en een heldere commit-structuur	Logische, goed omschreven commits die de ontwikkelingen helder in beeld brengt
Presentatie	Kan game demonstreren. Legt uit wat gemaakt is. Beantwoordt vragen.	Goede demo. Legt features uit. Bespreekt uitdagingen en oplossingen. Enthousiast.	Professionele presentatie. Excellente demo. Laat code zien. Inspireert anderen. Reflecteert op proces.

3.2 FOCUS PER WEEK

- Week 1 → Code structuur begrijpen, basis commando's maken
- Week 2 → Properties en methods, health systeem, items maken
- Week 3 → Collections (Dictionary), inventory systeem, complex logic
- Week 4 → Geavanceerde features, game polish, endgame condities
- Week 5 → Presenteren, reflecteren

3.3 CESUUR

Voor deze module dien je in ieder geval te voldoen aan de eisen zoals die hierboven vermeld staan.

3.4 AFRONDING

Zie week 5 bij 3.2 ☺

4. Planning en verantwoording

4.1 BESCHIKBARE TIJD

Voor deze module heb je 5 weken de tijd.
Contacttijd per week:

- 4× Instructieles
- 2× Verwerkingsles

5. Bronnen

Officiële Documentatie:

- Microsoft C# Documentation
- W3Schools C#

Veel succes en programmeerplezier!