

**FORSIDE FOR INNLEVERING**

**TILGJENGELIGHET**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Fri** | **x** | **Begrenset** |

|  |  |  |
| --- | --- | --- |
| **Emnekode og emnenavn:** | PG3300 Programvarearkitektur | |
| **Tittel norsk:** | NITH Standard Forside for Innleveringer | |
| **Tittel engelsk (Hovedprosjekt):** | null | |
| **Eventuell oppdragsgiver:** | null | |
| **Utleveringsdato:** | 8. oktober 2012 | |
| **Innleveringsdato:** | 11. november 2012 | |
| **Antall sider:** | 2 | |
| **Antall ord:** | 71 | |
| **Sammendrag (maks 100 ord):**  Dokumentet tar for seg en kort introduksjon av oppgaven og dokumentasjon av GUI, UI, og designvalg på kodenivå i innlevering 1 i emnet PG3300 Programvarearkitektur. Innholdet er delt inn etter deloppgaver. | | |
| **Gruppenummer:** | null | |
| **Studentnavn:** | **Studentnummer:** | **Signatur:** |
| Martin Lehmann | 700766 | null |
| Simen Bekkhus | 700532 | null |
| **Studentens signatur er også en bekreftelse av at hun/han har gjort seg kjent med, og fulgt, NITHs retningslinjer for intellektuell redelighet (tilgjengelig på intranett).** | | |

# Innholdsfortegnelse

[Innholdsfortegnelse 2](#_Toc340246244)

[Deloppgave 1 av 2: SnakeMess 3](#_Toc340246245)

[GUI 3](#_Toc340246246)

[UI 3](#_Toc340246247)

[Designvalg på kodenivå 3](#_Toc340246248)

[Objektorientering 3](#_Toc340246249)

[Deloppgave 2 av 2: Cookie Bakery 5](#_Toc340246250)

[GUI 5](#_Toc340246251)

[UI 5](#_Toc340246252)

[Designvalg 5](#_Toc340246253)

# Deloppgave 1 av 2: SnakeMess

## GUI

Alt av GUI var gitt i deloppgaven. Vi har gjort vårt ytterste for å gjenskape alt av visuelt innhold (inkludert at det blir generert et linjeskift hvis man «dør» helt nederst i høyre hjørne, selv om dette nok er å regne som en bug).

## UI

Brukeren kan benytte piltastene, som definert i oppgaven; WASD; eller NumPad 1 og 9 eller 3 og 7 for å navigere, tilsvarende det «originale» Snake-spillet. Brukeren kan også trykke Space for å pause spillet.

## Designvalg på kodenivå

### Objektorientering

Vi har splittet oppgaven inn i 16 klasser. Vi innser at dette er en ekstrem overdrivelse for dette lille prosjektet, men vi har skrevet programmet for at det skal være enkelt utvidbart til andre plattformer og/eller visninger i konsollen, i tillegg til andre systemer for å hente input.

### Beskrivelse av og begrunnelse for klasser

#### Generelt

##### SnakeGame

En abstrakt base for selve spillet. Definerer vindusstørrelse og framerate. For å utvide programmet, må det skrives nye klasser som arver fra SnakeGame.

##### ConsoleSakeGame

Arver fra SnakeGame, og tar seg av å sette opp spillet spesifikt for konsollen.

##### Program

Definerer hvilken plattform programmet skal kjøre på og starter «Game Loop»-en.

#### Spillobjekter (Game Objects)

##### DrawableGameObject

Alle tegnbare objekters mor; inneholder felles egenskaper for alle tegnbare objekter.

##### Snake

Selve slangen i form av en liste av alle kroppsdeler. Oppdaterer posisjonen til hver enkelt kroppsdel for å «flytte» slangen.

##### SnakeHead

Slangens hode. Har sin egen klasse i stedet for å være den samme klassen som SnakeBodyPart for å slippe å gjøre konsollspesifikke kall fra andre steder enn ConsoleView.

##### SnakeBodyPart

Hver enkelt av individuelle kroppsdeler. Har ansvar for posisjon, tekstur, og eventuell farge. Har sin egen klasse i stedet for å være en del av SnakeHead for å slippe å gjøre konsollspesifikke kall fra andre steder enn ConsoleView.

##### ConsoleBlank

Skriver over det bakerste elementet på slangen ved oppdatering for at slangen skal «bevege seg».

##### Loot

«Eplet» slangen skal «spise». Skrevet som en «singleton» for å optimalisere ytelse ved å flytte det samme objektet rundt i spillvinduet i stedet for å opprette et nytt objekt hver gang «eplet» blir «spist».

#### Arkitekturspesifikt

##### Model

Inneholder all data for spillet: i denne oppgaven (før eventuell utvidelse) Snake og Loot.

##### View

En abstrakt base for forskjellige typer visninger programmet kan ha. Før eventuell utvidelse er dette fullstendig meningsløst, men vi har valgt å løse det på denne måten for å gjøre programmet mer utvidbart, samt lettere å vedlikeholde. For å implementere flere visninger, må det skrives nye klasser som arver fra View.

##### ConsoleView

Arver fra View, og tar seg av den konsollspesifikke tegningen.

#### Verktøy (utilities)

##### InputHandler

Tar i mot og abstraherer faktisk input, og gir Input-klassen de abstraherte instruksjonene. For å utvide programmet til andre plattformer, må det skrives nye metoder tilsvarende CheckConsoleInput() : bool.

##### Input

Kontrollerer slangen basert på de abstraherte instruksjonene fra InputHandler.

##### Point

Todimensjonalt punkt for å definere posisjonen til alle tegnbare objekter, og å definere spillvinduets størrelse.

# Deloppgave 2 av 2: Cookie Bakery

## GUI

## UI

## Designvalg