# Resource Management

|  |  |
| --- | --- |
| **Læringsmål** | Du kan:   * **1Pf1:** anvende centrale metoder til at specificere og konstruere algoritmer [...] * **1Pf2:** anvende centrale faciliteter i programmeringssproget til realisering af algoritmer […] * **1Pf3**: anvende et i professionen udbredt, integreret udviklingsværktøj, herunder versionsstyringssystem […] til at designe og konstruere praksisnære applikationer […] * **1Pk3**: i en struktureret sammenhæng tilegne sig ny viden, færdigheder og kompetencer inden for programmeringssprog, udviklingsværktøjer, programmeringsteknikker og programdesign * **1Pf6**: anvende moderne teknikker og værktøjer til afvikling af test […] |
| **Forventet læringsudbytte [SOLO]** | Programmeringssporet:   * [Unistrukturel] Du kan huske enkelte kendetegn ved styring af **program flow** – throw, try-catch, try-finally, try-catch-finally * [Unistrukturel] Du kan huske enkelte kendetegn ved **datahåndtering** * [Unistrukturel] Du kan genkende enkelte kendetegn ved simple **C# datatyper** * [Unistrukturel] Du kan genkende enkelte kendetegn ved **C# klasse (objekt)** * [Unistrukturel] Du kan nævne enkelte kendetegn ved **C# interface** – naming, property, method * [Unistrukturel] Du kan nævne enkelte kendetegn ved **persistens** – File I/O, streams, tekstfil |
| **Din forberedelse** | Programmeringssporet:   * Lav informationssøgning omkring “relative vs absolute path” * Orientér dig i [File documentation](https://docs.microsoft.com/en-us/dotnet/api/system.io.file?view=netcore-3.1) * Genopfrisk interfaces: [C# Interfaces Explained in Simple Terms](https://www.youtube.com/watch?v=aQ8YkJrAbzE) (video: 5:03) * Intro til IDisposable og using:   + [File.Open and .Close](https://www.linkedin.com/learning/object-oriented-programming-with-c-sharp/file-open-and-close?u=57075649) [5:18]   + [File.Close in the finally block](https://www.linkedin.com/learning/object-oriented-programming-with-c-sharp/file-close-in-the-finally-block?u=57075649) [2:45]   + [Keyword using](https://www.linkedin.com/learning/object-oriented-programming-with-c-sharp/keyword-using?u=57075649) [5:44] * Se følgende [simple using eksempel](https://www.linkedin.com/learning/c-sharp-dot-net-programming/a-simple-example?u=57075649) * Læs om [Using objects that implement IDisposable](https://docs.microsoft.com/en-us/dotnet/standard/garbage-collection/using-objects)   + Fokus på, hvordan StreamReader er implementeret internt ved brug af using (try-finally) * Se på Tusindfryd analyse-artefakterne *igen* (OM, DM, UC, SSD, OC) med fokus på systemsekvensdiagrammet ”Start produktion” |

# Dagens citat:

Understanding how to apply GRASP for object design is a key goal of the book.   
([Larman], side 277)

# Øvelse 1: Terminologi

Nu skal du teste din forforståelse af dagens emne(r) med udgangspunkt i dagens forberedelse.

Del teamet op i to mindre grupper, og brug **Møde på midten** i hver gruppe til at diskutere begreberne ”File”, ”Interface”, ”IDisposable” og ”using”. Husk, at alle i gruppen skal have taletid.

# Øvelse 2: IDisposable

I denne øvelse skal du og dit team se nærmere på interfaces. Mere specifikt skal du se nærmere på using, der implementerer interfacet *IDisposable*.

Benyt parprogrammering.

## Øvelse 2.1: try- finally StreamReader

Udfør følgende:

* Opret en konsolapplikation
* Læs en linje fra en tekst-fil (.txt)
* Benyt StreamReader (Hint: Se i Ex11-Persistence for inspiration)
* Benyt try-finally, hvor du lukker StreamReader forbindelsen i finally

## Øvelse 2.2: Refactoring

I denne øvelse skal du lave et genvisit af Ex11-Persistence. Du skal benytte dig af using i hele din løsning. Husk, at jeres UnitTests skal stadigvæk lyse grønt.

# Øvelse 3: Tusindfryd

I dag skal du arbejde videre med Tusindfryd-casen. Du skal som sidste gang igen se på at udforme et SD ud fra en OC – og derefter kode ud fra jeres SD.

Hvis du og dit team ikke blev færdige sidste gang, start da, hvor I slap. Benyt parprogrammering.

## Øvelse 3.1: SD ud fra OC

Du og dit team skal nu udforme et sekvensdiagram (SD) samt tilpasse DCD’et ud fra operationen *startProduktion*; hertil har du fået udleveret følgende operationskontrakt (OC):

|  |  |
| --- | --- |
| **Operation:** | startProduktion(drivhus:string, produktionsbakke:string, blomstersort:string, startAntal:integer, startDato:DateTime) |
| **Cross References:** | Start produktion |
| **Preconditions:** | * produktionsbakke findes * blomstersort findes |
| **Postconditions:** | * et produktionsobjekt med startAntal og startDato og forventetSlutantal blev oprettet (instance creation) * produktionsbakke objekt blev associeret til produktionsobjektet (association formed) * blomstersort objekt blev associeret til produktionsobjektet (association formed) |

Sørg for, at dine kvalitetskriterier for SD og DCD bliver overholdt.

## Øvelse 3.2: Kvalitetssikring af SD

Når du og dit team er færdige med at udforme jeres SD, skal I have kvalitetssikret jeres artefakt med minimum et andet team.

Lav rettelser til jeres artefakt og kvalitetskriterier ud fra behov.

## Øvelse 3.3: Implementering af kode ud fra SD

Fortsæt, hvor du og dit team slap sidst. Hvis I blev færdige med implementeringen af det første SD, så skal I fortsætte med SD’et fra i dag. Det er tilladt at oprette ’fake data’.

**Bemærk:** Du og din sidemakker må gerne spørge dine medstuderende/os undervisere/lektiecafé-hjælperne til råds, hvis I er gået i stå.