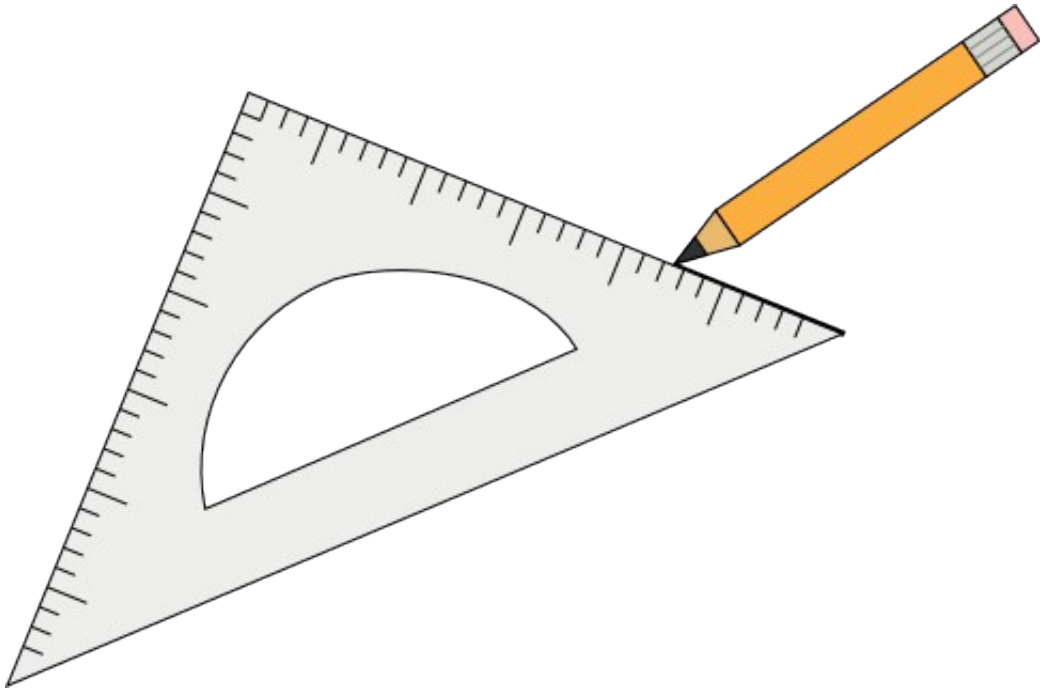
Standarder



Version:

1.0: Lavet basis ud fra gamle semester projekt med rettelser for tilpasning.

1.1: Revision med hensyn til sprog på billeder samt underscore i parametre.

1.2: Ændret punkt om revisionkontrol fra SVN til Git, samt lavet om i formaten for en commit med hensyn til milestone.

Beskrivelse:

Indeholder liste over forskellige standarder, som skal følges i udviklingen af Sorting Industrial Robot(SIR).

Værktøjer:

- Tekst dokumentation: Værktøj der kan håndtere Open Document Format(.odt). (Helst LibreOffice)
- Kode: Visual Studio. Kan også bruge andre tekst editors under udvikling, men alle projekt filer skal være en Visual Studio solution.
- Diagrammer:
    - Klassediagrammer: Visio
    - Sekvensdiagrammer: Visio

      Ellers frit. Skal bare som standard eksporteres til minimum PNG format.
- Kommunikation: Skype og TeamViewer.
- Testning: Til testning af kode bruges NUnit og dens tilhørende syntakst samt dotCover.

Dokumenter:

- Tekst:
    - Font Arial størrelse 12. Headings er standard headings(Maksimum 3) med størrelserne:
        - Heading 1: 16
        - Heading 2: 14
        - Heading 3: 12

          Alle er **BOLD**.
    - Linjeafstand: 1.5.
- Dokument format: Dokumenter skal være i Open Document Format(.odt) samt PDF.
- Aflevering: Dokumenter der skal afleveres skal være i Portable Document Format(.pdf).

Billeder og diagrammer:

- Sprog: Sprog brugt på billeder og diagrammer skal være på engelsk.

Kode:

- Klasse navne: Tages direkte fra klassediagrammer.
- Variabel navne: Hungarian plus Camelcase.(stringUserName)
  Note: Parameter navne skal starte med underscore '_'.
- Funktions navne: Camelcase.(såNogetSomDetHer())
- Filnavne:
  - Hvis indeholder enkelt klasse har den navn efter klassen(Lille begyndelses bogstav) ellers skal den have et sigende navn, for de ting den indeholder.
- Hver fil skal have en header med minimum:
  - Description: Kort beskrivelse af hvad filen indeholder og hvad bruges til.
  - Author: Forfatteren Sorting Industrial Robot(SIR).
- I kildefilerne skal hver funktion have minimum:
  - Beskrivelse af argumenter. Eks.:
    - \param x1 Beskrivelse; eller /// <param name="x1">beskrivelse</param>
  - Beskrivelse af retur type: Eks.:
    - \return Beskrivelse; eller /// <returns>Beskrivelse</returns>
- Sprog: Engelsk for klasse, variable og funktions navne samt kommentarer.
- Formatering:
  - Indryk efter funktioner, klasser og structs, før der bliver brugt '{'.
    Ex:

```
void function1()
{
        …
```

  - Mellemrum for tabs: 4.

Version Control System:

- Type: Git
- Commits:
  - 'Headeren'(Altså den første linje):

    <BUG><Status:GREEN/RED>: Kort beskrivelse

    - Bug: Hvis en bug er fundet. Beskriv i mere detalje senere.
    - Status: Beskriver om **ALLE** unit tests har kunne køre succesfuldt. Skal kun være der, hvis der er blevet rodet med kode filer. Dette gælder også refactoring.
  - Indholdet(Alt efter første linje):
    - Hvad lavet: <Punktliste god idé>. Ex.:
      - GUI: Indsat knap til at fjerne tabel værdi.
      - CalcLogic: Forbedret farten på udregning af massefylde.
    - Bugs: Hvis bugs fundet.
    - Andet: Ting ellers af interesse.
  - Andet:
    - Tagging: Hvis noget 'større' gjort færdigt, for eksempel en use case eller en release, skal committen tagges og pushes til origin. Dette gør det meget nemmere senere at komme tilbage til dette punkt.

Testning:

- Navne på funktioner og klasser: Tjek ”\Dokumentation\Andre Billeder\testingNames” Som er taget fra bogen ”The Art of Unit Testing”.