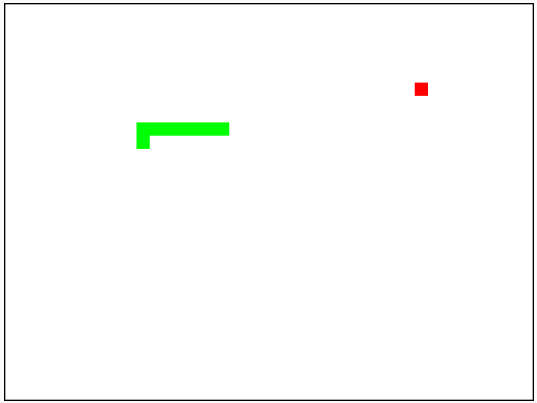
Phase I - Modellierung eines Spiels

Modellierung - 10min

- Erstelle ein Klassendiagramm mit allen **Klassen**, die du im linken Screenshot ausmachen kannst.
- Notiere die **Attribute**, die du sehen kannst und die, die Klassen haben müssen.
- Notiere die **Methoden**, die die Klassen vermutlich haben.
- Zeichne und beschrifte die Beziehungen zwischen den Klassen.



“Offensichtliche” Klassen

Schlange

Attribute:

- `richtung` *(rauf, runter, links, rechts)*
- `koerperteile`

Methoden:

- bewegen
- wachse

Essen

- Position

Schlange
richtung koerperteile
bewegen() wachse()

Essen
Position

Arbeitsauftrag - 15 min

- Kopiere die **Snake** BlueJ-Projekt Vorlage
- Erzeuge Objekte der bereits vorhandenen Klassen und untersuche ihre Methoden
- Erstelle eine neue Klasse `Schlange`
  - Ergänze die Klasse Schlange um das Attribut `String richtung`
  - Ergänze die Klasse Schlange um ein Attribut `kopf` vom Typ `Rechteck`
  - Implementiere den `Konstruktor` der Klasse `Schlange`
    - Es soll die Position übergeben werden
    - Die Startrichtung soll `"runter"` sein
    - Erzeuge einen neuen grünen Kopf an der richtigen Position mit der Größe 20x20.

Arbeitsauftrag - 10 min

- Implementiere die Methode `bewegen` der Schlange
- Je nach ihrer Richtung soll sich der Kopf der Schlange um 20 Pixel bewegen
  Tipp: verwende die Methode `Verschieben(int deltaX, int deltaY)` der Klasse `Rechteck`
- Erzeuge ein Schlangen Objekt und verschiebe es über den Bildschirm.

Wer ruft die Methoden der Klasse Schlange auf?

Wer ruft die Methoden der Klasse Schlange auf?

Once class to rule them all

Eine Klasse verwaltet das gesamte Spiel indem sie auf Nutzereingaben reagiert und die richtigen Methoden der anderen Objekte ausführt.

Welche Attribute hat diese Klasse?

Wer ruft die Methoden der Klasse Schlange auf?

Once class to rule them all

Eine Klasse verwaltet das gesamte Spiel indem sie auf Nutzereingaben reagiert und die richtigen Methoden der anderen Objekte ausführt.

💡 Welche Attribute hat diese Klasse?

Attribute

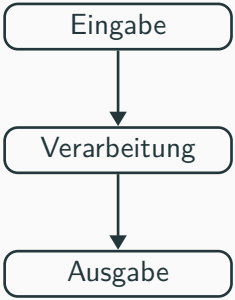
- schlange
- essen
- rahmen
- zeit
- (highscore)

EVA-Prinzip

EVA-Prinzip

Trennung der Datenverarbeitung in:

- **E**ingabe: `input()`
- **V**erarbeitung: `update()`
- **A**usgabe: `output()`

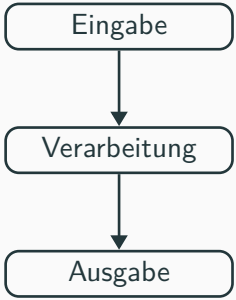


EVA-Prinzip

Trennung der Datenverarbeitung in:

- **E**ingabe: `input()`
- **V**erarbeitung: `update()`
- **A**usgabe: `output()`

Wie oft müssen wir die drei Schritte ausführen?



EVA-Prinzip

Trennung der Datenverarbeitung in:

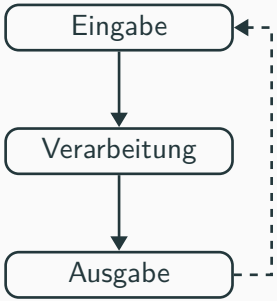
- **E**ingabe: `input()`
- **V**erarbeitung: `update()`
- **A**usgabe: `output()`

Wie oft müssen wir die drei Schritte ausführen?

Solange das Spiel läuft!

EVA-Loop

```
solange Spiel aktiv
 input()
 update()
 output()
```



Sind wir die ersten mit diesem Problem?

Spieleengine

Sind wir die ersten mit diesem Problem?

Sicher nicht! Also lasst uns das Rad nicht neu erfinden. Sondern eine **Spieleengine** verwenden die uns die lästigen Dinge abnimmt.

Spieleengine

Sind wir die ersten mit diesem Problem?

Sicher nicht! Also lasst uns das Rad nicht neu erfinden. Sondern eine **Spieleengine** verwenden die uns die lästigen Dinge abnimmt.

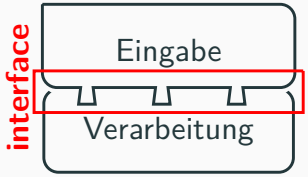
GraphicsAndGames

```
solange Spiel aktiv <- GraphicsAndGames
  input()           <- GraphicsandGames
  update()          <- Wir
  output()          <- GraphicsAndGames
```

Wie bekommen wir die Eingabe?

Wir erfüllen mit unserer Spiel-Klasse das **interface** `Zeichenfenster.AktionsEmpfaenger` damit uns GraphicsAndGames Eingaben übergeben kann.

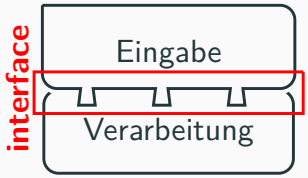
Ein Interface gibt zu implementierende Methoden vor.



Wie bekommen wir die Eingabe?

Wir erfüllen mit unserer Spiel-Klasse das **interface** `Zeichenfenster.AktionsEmpfaenger` damit uns GraphicsAndGames Eingaben übergeben kann.

Ein Interface gibt zu implementierende Methoden vor.



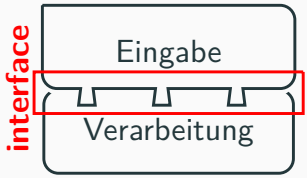
Java

```
class Snake implements Zeichenfenster.AktionsEmpfaenger
```

Wie bekommen wir die Eingabe?

Wir erfüllen mit unserer Spiel-Klasse das **interface** `Zeichenfenster.AktionsEmpfaenger` damit uns GraphicsAndGames Eingaben übergeben kann.

Ein Interface gibt zu implementierende Methoden vor.



Java

```
class Snake implements Zeichenfenster.AktionsEmpfaenger
```

Methoden

- `public void Ausführen()`
- `public void Taste(char taste)`
- `public void SonderTaste(int taste)`
- `public void Geklickt(int x, int y, int anzahl)`

Arbeitsauftrag

- Erstelle eine neue Klasse `Snake`
  - Ergänze die Klasse `Snake` um ein **Referenzattribut** `schlange`
  - Initialisiere das Attribut `schlange` im `Snake` Konstruktor an der Position (380, 280).
  - Ergänze die Klasse `Snake` um die 4 Methoden des `AktionsEmpfaenger` interfaces.
  - Definiere, dass die Klasse `Snake` das interface `AktionsEmpfaenger` erfüllt.
  - Registriere neue `Snake`-Objekte als AktionsEmpfanger, indem du den Konstruktor um den Aufruf `Zeichenfenster.AktionsEmpfängerEintragen(this);` erweiterst.

Arbeitsauftrag

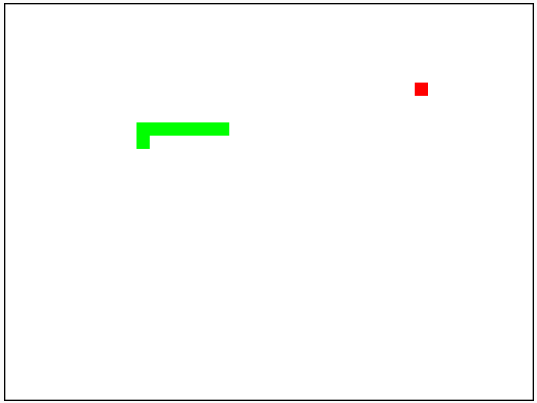
- Implementiere die Methode `Taste(char taste)` sodass die Richtung der Schlange angepasst wird.
  - 'W': rauf
  - 'S': runter
  - 'A': links
  - 'D': rechts
- Rufe `schlange.bewegen()` in der `Ausführen` Methode auf
- Teste dein Spiel indem du ein neues `Snake` Objekt erzeugst
- **Für Schnelle:** Implementiert die Schlangensteuerung mit den Pfeiltasten und der Methode `Sondertaste` (Tipp: die Klasse `java.awt.event.KeyEvent` hilft)

Phase II - Essen und Wachstum

Essen

Essen
Position

Das Essen soll in festen Zeitintervallen an einer zufälligen Position entstehen.



Zufall

Die Java Klasse `java.util.Random` stellt *pseudo* zufällige Zahlen bereit.

Zufall

Die Java Klasse `java.util.Random` stellt *pseudo* zufällige Zahlen bereit.

Verwendung

- Importieren der Klasse: `import java.util.Random;`
- Deklarieren und Initialisieren eins Attributs vom Typ `Zufall`.
- Die Methode `nextInt(int max)` liefert eine Zahl zwischen 0 und `max`.

In welchem Bereich soll das Essen platziert werden?

Zufall

Die Java Klasse `java.util.Random` stellt *pseudo* zufällige Zahlen bereit.

Verwendung

- Importieren der Klasse: `import java.util.Random;`
- Deklarieren und Initialisieren eins Attributs vom Typ `Zufall`.
- Die Methode `nextInt(int max)` liefert eine Zahl zwischen 0 und `max`.

In welchem Bereich soll das Essen platziert werden?

Größe des Welt

Das Zeichenfenster misst 800x600 Pixel.

Tischlein deck dich

```
void Ausführen() {
  ticks++;
  if (ticks % 30 == 0) {
    if (essen) {
      essen.entfernen();
    }
    int x = zufall.nextInt(780);
    int y = zufall.nextInt(580);
    essen = new Essen(x, y);
  }
  ...
}
```

Wachstum

Die Schlange soll im Laufe des Spiels länger werden.

Welcher Datentyp eignet sich dazu eine unbekannte Anzahl an Rechtecken zuspeichern?

Wachstum

Die Schlange soll im Laufe des Spiels länger werden.

💡 Welcher Datentyp eignet sich dazu eine unbekannte Anzahl an Rechtecken zuspeichern?

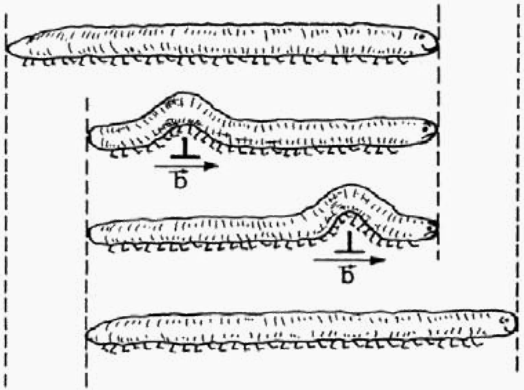
Das Attribut `koerper`

- Das Attribut `koerper` der Klasse `Schlange` vom Typ `Rechteck[]` verwaltet alle Körperteile
- Felder haben eine **feste** Länge, deshalb muss jedes mal ein neues größeres Feld erzeugt werden und die einzelnen Rechtecke von dem alten Feld in das neue Kopiert werden.

Bewegung

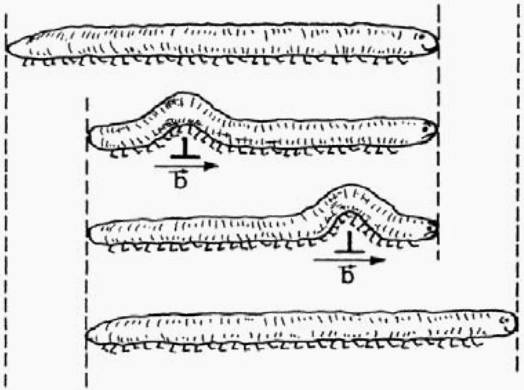
Wie kommt die Schlange um die Ecke?

Wie kommt die Schlange um die Ecke?



Wie kommt die Schlange um die Ecke?

Jedes Körperteil folgt seinem Vorgänger.



Bewegung

```
void bewegen() {
  for (int i = koerper.length - 1; i > 0; i--) {
    Rechteck vorgaenger = koerper[i-1];
    koerper[i].PositionSetzen(vorgaenger.x, vorgaenger.y);
  }

  Rechteck kopf = koerper[0];
  // Kopf je nach aktueller Richtung verschieben
  // kopf.Verschieben()
  ...
}
```

Phase III - Kollisionserkennung

Wann frisst die Schlange das Essen

Berührt der Kopf das Essen?

Wann frisst die Schlange das Essen

Wo findet die Überprüfung statt?

Wann frisst die Schlange das Essen

Wo findet die Überprüfung statt?

In der `Ausführen`-Methode der Welt-Klasse `Snake`, da

- dort Schlange und Essen bekannt sind

Wann frisst die Schlange das Essen

Wo findet die Überprüfung statt?

In der `Ausführen`-Methode der Welt-Klasse `Snake`, da

- dort Schlange und Essen bekannt sind
- dort die Schlange bewegt wird

Wann frisst die Schlange das Essen

Wo findet die Überprüfung statt?

In der `Ausführen`-Methode der Welt-Klasse `Snake`, da

- dort Schlange und Essen bekannt sind
- dort die Schlange bewegt wird

Wann frisst die Schlange das Essen

💡 Wo findet die Überprüfung statt?

In der `Ausführen`-Methode der Welt-Klasse `Snake`, da

- dort Schlange und Essen bekannt sind
- dort die Schlange bewegt wird

Pseudocode

```
Schlange bewegen
Falls Kopf das Essen berührt
  entferne Essen
  Schlange wachse
```

Kollision mit dem Rand der Welt

Ist die Position des Kopfes noch Teil der Welt?

Kollision mit dem Rand der Welt

Wo findet die Überprüfung statt?

Kollision mit dem Rand der Welt

Wo findet die Überprüfung statt?

In der `bewegen`-Methode der Klasse `Schlange`, nachdem der Kopf bewegt wurde

Woher weiß das Spiel das die Schlange kollidiert ist?

Kollision mit dem Rand der Welt

Wo findet die Überprüfung statt?

In der `bewegen`-Methode der Klasse `Schlange`, nachdem der Kopf bewegt wurde

Woher weiß das Spiel das die Schlange kollidiert ist?

Die Methode `bewegen` gibt einen Wahrheitswert zurück, ob sie erfolgreich war.

Kollision mit dem Rand der Welt

Wo findet die Überprüfung statt?

In der `bewegen`-Methode der Klasse `Schlange`, nachdem der Kopf bewegt wurde

Woher weiß das Spiel das die Schlange kollidiert ist?

Die Methode `bewegen` gibt einen Wahrheitswert zurück, ob sie erfolgreich war.

Pseudocode

```
Koerper bewegen
Kopf bewegen
Falls kopf.x < 0 oder kopf.x > Weltbreite
      oder kopf.y < 0 oder kopf.y > Welthöhe
  return false
return true
```

Kollision mit sich selbst

Berührt der Kopf der Schlange eins der eigenen Körperteile?

Kollision mit sich selbst

Wo findet die Überprüfung statt?

Kollision mit sich selbst

Wo findet die Überprüfung statt?

In der `bewegen`-Methode der Klasse `Schlange`, nachdem die Schlange bewegt wurde.

Kollision mit sich selbst

Wo findet die Überprüfung statt?

In der `bewegen`-Methode der Klasse `Schlange`, nachdem die Schlange bewegt wurde.

Pseudocode

```
Koerper bewegen
Kopf bewegen
...

Für jedes koerperteil hinter dem kopf
  Falls kopf beruehrt koerperteil
    return false

return true
```