Zusammenfassung - struct und enum!

Aufbau:

```
struct Person {

// Eigenschaften

    // init
// Der memberwise init

    // Methoden
// mutating Methoden

}
```

Nutzung / Merkmale:
- Eigenen Typen erstellen
- Daten werden kaum verändert
- Daten ablegen in einem Typ
- Keine Vererbung nötig
- Value(Wert) Typ

Aufbau:

```
class Person {

// Eigenschaften

    // init

    // Methoden

}
```

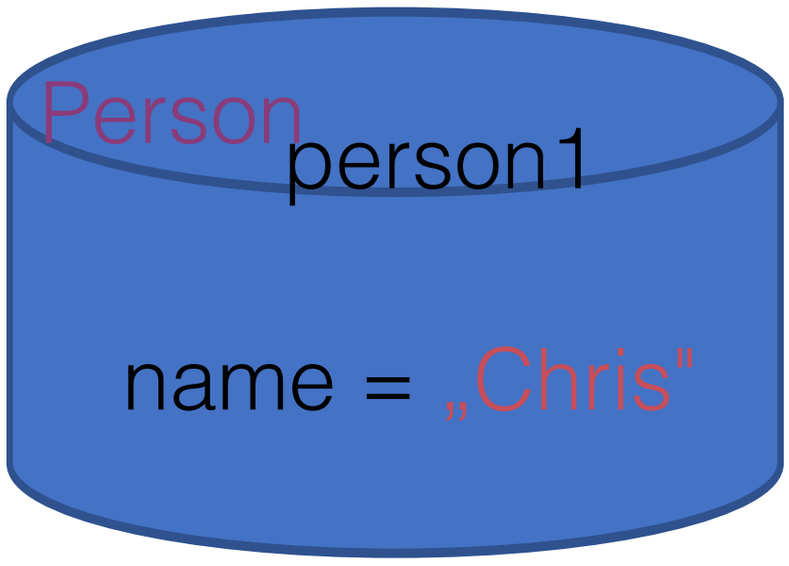
Nutzung / Merkmale:
- Eigenen Typen erstellen
- Viele Daten
- Daten werden ständig geändert
- Viele Methoden notwendig
- Vererbung notwendig
- Reference (Adressen) Typ

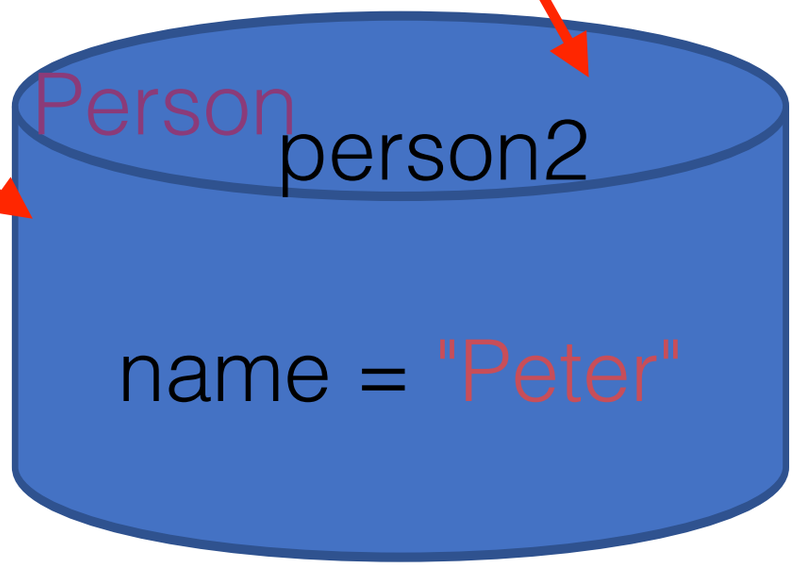
Klassen = Referenz Typ (Adressen Typ)

```
var person1 = Person(name: "Chris")
person1.name
```

```
var person2 = Person(name: "Peter")
person2.name
```

person1 = person2



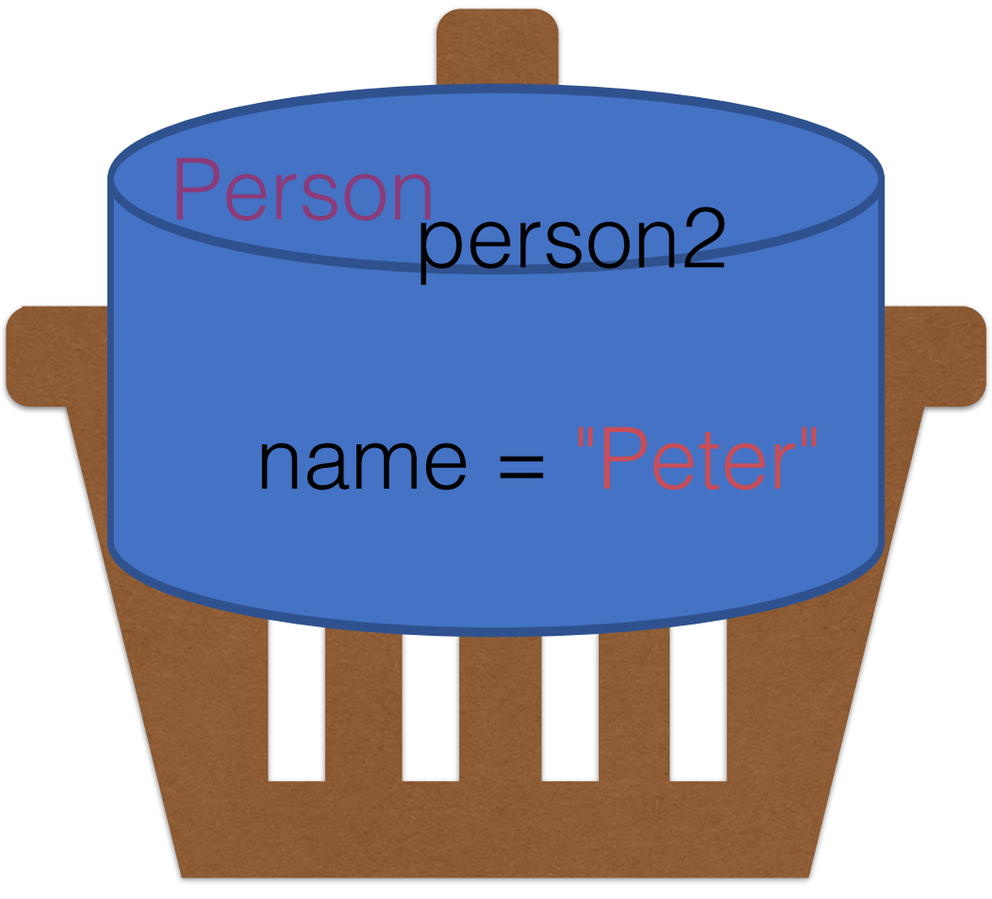


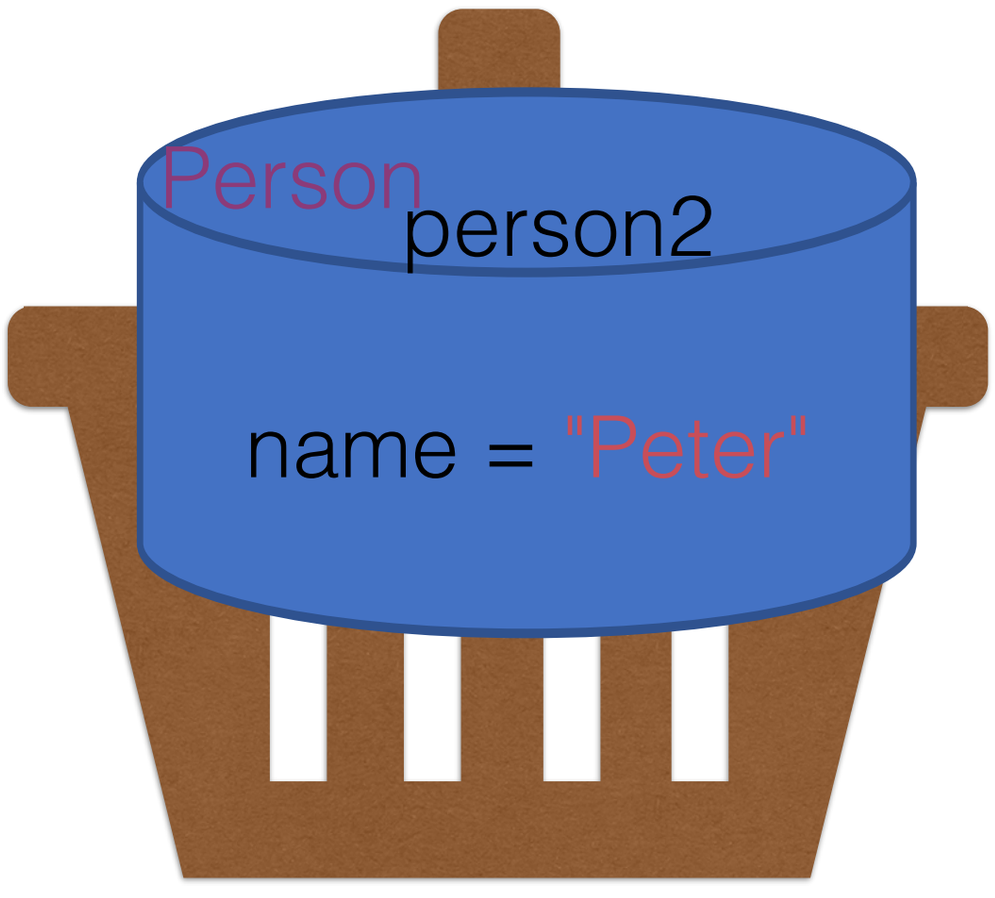
struct = Value Typ (Wert Typ)

```swift
var person1 = Person(name: "Chris")
person1.name
```

```swift
var person2 = Person(name: "Peter")
person2.name
```

person1 = person2





value Typ = der Ganze Container ist dort hinterlegt

Zusammenfassung - struct und enum!

Aufbau:

```
enum Playerlevel {
  case anfänger
  case fortgeschritten
  case profi
}
```

```
enum Kompass {
  case nord
  case süd
  case ost
  case west
}
```

```
enum Constans: Double {
  case pi = 3.14159
  case e = 2.718
  case phi = 1.618

}
```

Nutzung / Merkmale:
- Eigenen Typen erstellen
- Ähnliche Fälle gruppieren
- Typ Sicherheit ->
da Variable vom Typ eines enum nur diese Fälle annehmen kann

Merke:
Datentyp gibt an was in einer Variable gespeichert werden kann

- Standart werte heißen raw values
  - raw vaules müssen von gleichen Typ sein

  - Folgende Typen sind erlaubt: Int, Float, String, Boolean -> gilt nur bei raw values