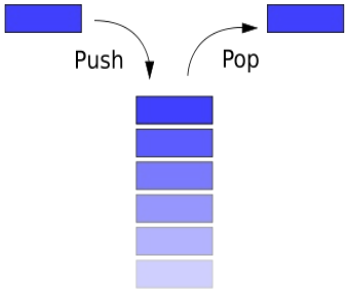
Stack (=Stapel)



Funktionsweise
Ein Stapel (Stack) kann eine beliebige Menge von Objekten aufnehmen und gibt diese, in entgegengesetzter Reihenfolge zur Aufnahme, wieder zurück.

Veranschaulichung: Stapel von Spielkarten
Einen Stack kann man sich als einen Stapel von Spielkarten vorstellen. Für den Kartenstapel sind drei Handlungen möglich:

- Die oberste Karte anschauen: Methode **`top()`**
- Die oberste Karte wegnehmen: Methode **`pop()`**
- Auf den Stapel eine Karte drauflegen: Methode **`push(neueKarte)`**

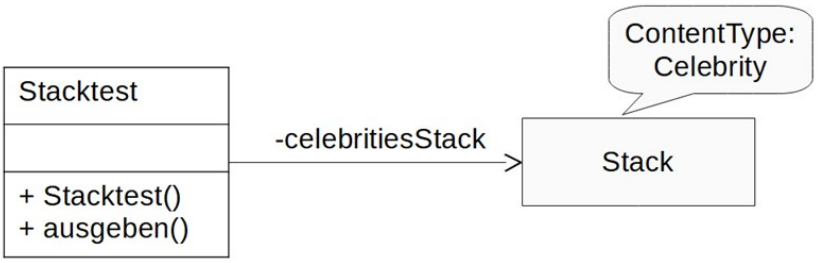
Vergleich Stack - Array
Gegenüber Arrays haben Stacks den Vorteil, dass sie beliebig viele Elemente aufnehmen können! Arrays dagegen werden zu Anfang mit einer bestimmten Größe initialisiert.
Der Nachteil von Stacks besteht darin, dass man immer nur das oberste Element ansprechen kann. Deswegen eignen sie sich nicht für alle Szenarien.

Stacks sind dynamische Datenstrukturen
- In Stacks werden die Objekte erst zur Laufzeit (=dynamisch) hinzugefügt.
- Die im Stack enthaltene Datenmenge kann so dynamisch wachsen oder schrumpfen. Die Anzahl der Knoten ist höchstens durch den vorhandenen Speicherplatz beschränkt.
- Arrays sind ***keine*** dynamischen Datenstrukturen, denn ihre Größe wird direkt zu Anfang festgelegt und kann zur Laufzeit nicht geändert werden.

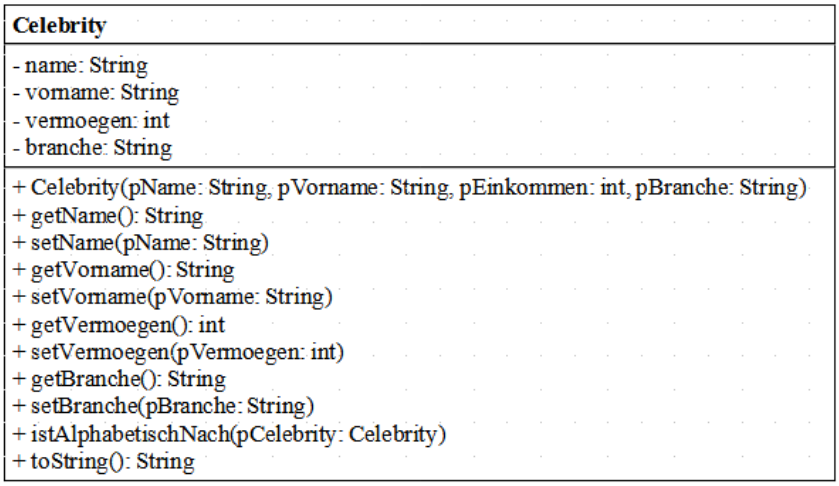
Dokumentation der Klasse `Stack`

Konstruktor	**`Stack<ContentType>()`** Ein leerer Stapel ist erzeugt. Objekte, die in diesem Stapel verwaltet werden, müssen vom Typ `ContentType` sein. *`ContentType` kann z.B. `String` sein, aber auch z.B. `Person`.*
Anfrage	**`isEmpty(): boolean`** Die Anfrage liefert den Wert ***`true`***, wenn der Stapel keine Elemente enthält, sonst liefert sie den Wert ***`false`***.
Auftrag	**`push (ContentType pContent)`** Das Objekt `pContent` wird oben auf den Stapel gelegt. Falls `pContent` gleich `null` ist, bleibt der Stapel unverändert.
Auftrag	**`pop()`** Das zuletzt eingefügte Element wird von dem Stapel entfernt. Falls der Stapel leer ist, bleibt er unverändert.
Anfrage	**`top(): ContentType`** Die Anfrage liefert das .oberste Stapelobjekt. Der Stapel bleibt unverändert. Falls der Stapel leer ist, wird `null` zurückgegeben.

In dem Stack **`celebritiesStack`** sind Objekte vom Typ **`Celebrity`** gespeichert. Für `celebritiesStack` gilt folgendes Implementationsdiagramm.



Chalamet, Timothee, 25
Lawrence, Jennifer, 140
Cruise, Tom, 891
Swift, Taylor, 150
Helene, Fischer, 13
Lame, Khaby, 12
Reif, Pamela, 9
Rowling, JK, 920
Vettel, Sebastian, 250
Gauff, Coco, 27
Wirtz, Florian, 19
Schroeder, Dennis, 60



Aufgaben

a) Erläutere zeilenweise die Methode `gibWas`.

b) Gib der lokalen Variable `c` einen sinnvollen Namen.

c) Die Methode wird wie folgt aufgerufen:
`boolean b = gibWas("Cruise");`
- Welchen Wert hat `b` nach dem Aufruf der Methode?
- Wie sieht `celebritiesStack` nach Ablauf der Methode aus?

d) Welche Aufgabe erfüllt die Methode?
Benenne sie entsprechend.

```
public boolean gibWas (String pName){
  boolean ergebnis = false;
  while(celebritiesStack.isEmpty() == false){
    Celebrity c = celebritiesStack.top();
    if(c.getName().equals(pName)){
      ergebnis = true;
    }
    celebritiesStack.pop();
  }
  return ergebnis;
}
```

Hinweis: Strings muss man mit dem Befehl `equals(...)` vergleichen.