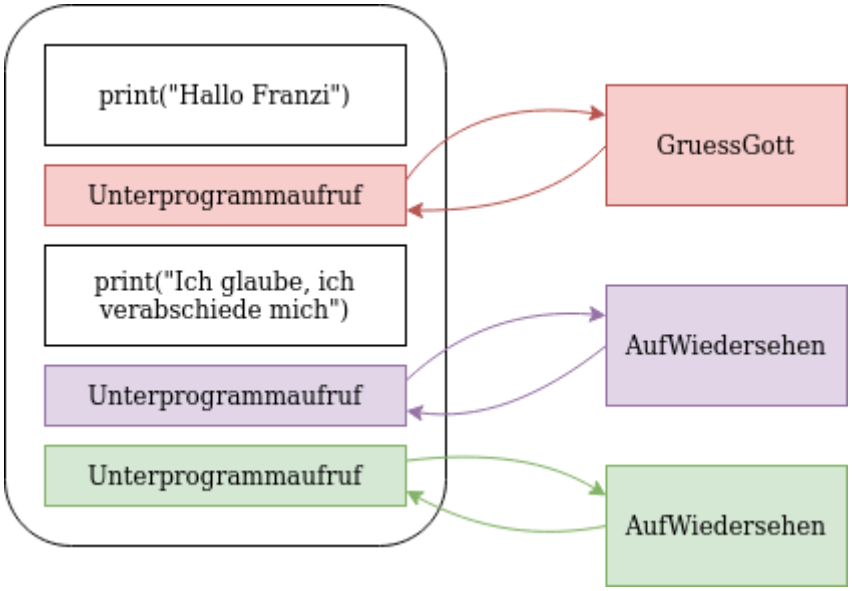
Der Programmaufrufstack

Unter dem (Programm)Aufrufstapel (englisch call stack, procedure stack) versteht man einen besonders genutzten Stapelspeicher, der zur Laufzeit eines Programms den Zustand der gerade aufgerufenen Unterprogramme enthält.

Was bedeutet das? Wir betrachten ein Beispiel:

```
public class Hallo
{
    public Hallo(String name)
    {
        System.out.println("Hallo " + name + "!");
        GruessGott(name);
        System.out.println("Ich glaube, gleich verabschiede ich mich...");
        AufWiedersehen();
        AufWiedersehen();
    }

    public void GruessGott(String name) {
        System.out.println("Grueß Gott, " + name + "!");

    }

    public void AufWiedersehen() {
        System.out.println("Ciao!");
    }

}
```

Beim Aufruf des Konstruktors werden also zwei weitere Methoden, also "Unterprogramme", aufgerufen, der Ablauf sieht etwa wie folgt aus.



Dabei fällt auf, dass der Programmablauf nach jedem Aufruf eines Unterprogramms zu der Stelle zurückkehren muss, von der dieses aufgerufen wurde. Das wird dadurch realisiert, dass beim Aufruf einer Methode/Funktion auf dem Programmaufrufstack für jede Methode ein Speicherbereich reserviert wird, in dem alle relevanten Infos festgehalten werden, unter anderem auch die Rücksprungadresse, also die Stelle, von der aus die Methode gestartet wurde.

Was passiert	**Wie sieht der Stack aus?**
Der Konstruktor wird mit dem Argument `name` aufgerufen. Auf dem Stack wird Speicher für den \\Konstruktoraufruf reserviert. Es gibt keine Rücksprungadresse	Konstruktor name = "Franzi" | ruecksprungadresse=keine
Aus dem Konstruktor heraus wird die Methode `GruessGott` mit dem Argument `name` aufgerufen. Die Stelle für die Rückkehr nach Bearbeitung von `GruessGott` wird vermerkt.	GruessGott name = "Franzi" | ruecksprungadresse = 0001 Konstruktor name = "Franzi" | ruecksprungadresse=keine

From:
https://www.info-bw.de/ -

Permanent link:
https://www.info-bw.de/faecher:informatik:oberstufe:algorithmen:rekursion:programmaufrufstack:start?rev=1642063726

Last update: **13.01.2022 08:48**