Il ciclo di vita della Servlet

Le fasi

Il servlet container controlla il ciclo di vita di una servlet.

Se non esiste una istanza della servlet nel container:

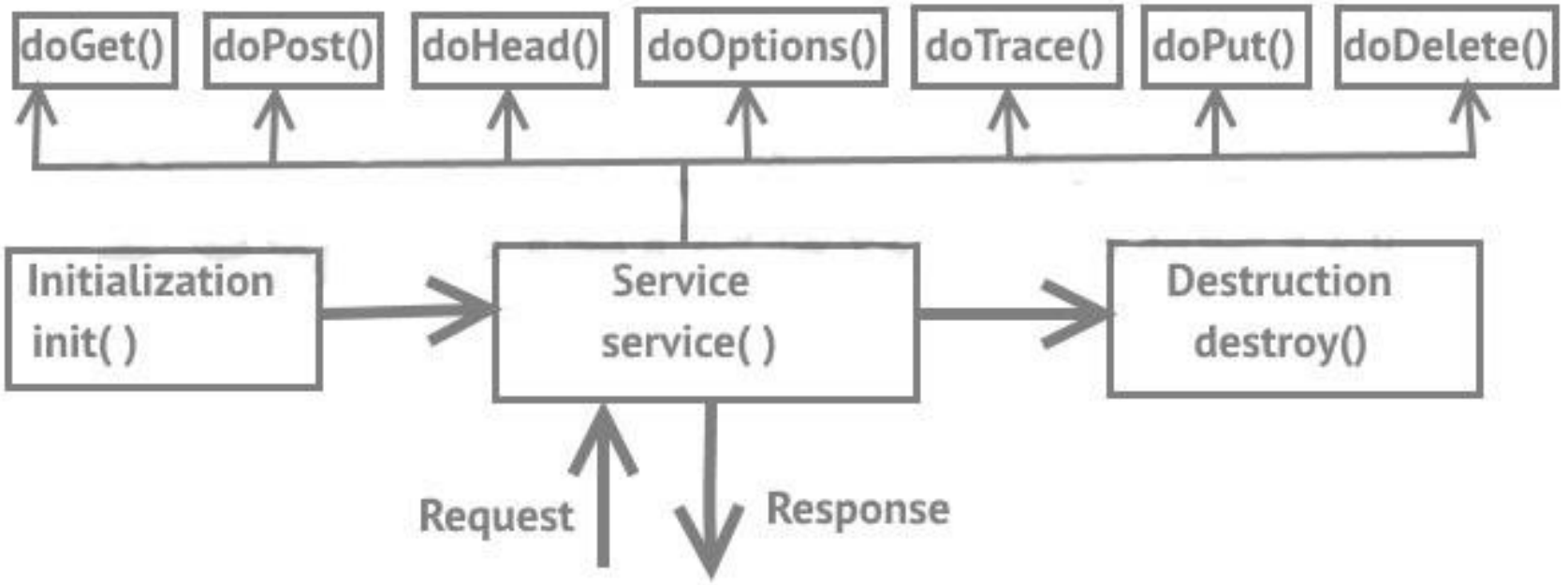
1) Carica la classe della servlet
2) Crea una **istanza** della servlet
3) Inizializza la servlet (invoca il metodo **init()**)

Poi invoca la servlet

4) Metodi **doGet()** o **doPost()** a seconda del tipo di richiesta ricevuta

Infine distrugge la servlet

5) Metodo **destroy()**



I metodi: init()

Viene chiamato una sola volta al caricamento della servlet.

È in questo metodo che la servlet **istanzia** tutte **le risorse** che utilizzerà poi per gestire le richieste: ad esempio si crea la connessione con un database

Esempio:

```
public void init(ServletConfig config) throws ServletException {
    super.init(config);
    String initial = config.getInitParameter("initial");
    try {
      count = Integer.parseInt(initial);
    }
    catch (NumberFormatException e) {
      count = 0;
    }
}
```

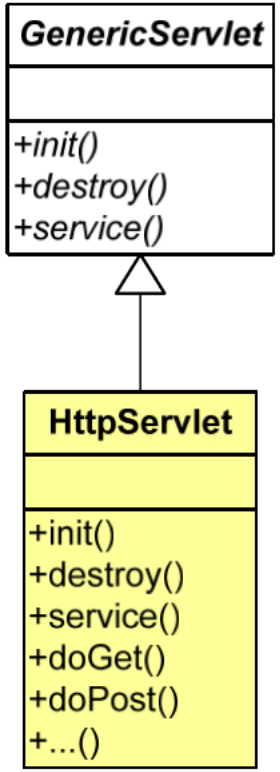
I metodi: service()

Nella classe **GenericServlet**, il metodo è astratto.

GenericServlet è estesa da **HTTPServlet** che fornisce una implementazione del metodo service()

- Delega l'elaborazione della richiesta ai metodi **doGet(), doPost(), doPut()** o **doDelete()** a seconda del tipo di HTTP Request ricevuta.

- Passa come parametri due oggetti di tipo **HttpServletRequest** ed **HttpServletResponse**



Esempio di servlet

```
package controller;

import java.io.IOException;
...

public class Student extends HttpServlet{

    public Student() {
        ...
    }

    public void init(){
        ...
    }

    public void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        ...
    }

    public void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        ...
    }

    public void destroy(){
        ...
    }

}
```

I metodi: destroy()

Viene chiamato una sola volta quando la servlet deve essere disattivata (es. quando deve essere rimossa).

Tipicamente serve per rilasciare le risorse acquisite (es. connessione ad un database).

Questo metodo segna la chiusura della servlet, è qui che si effettuano eventuali salvataggi di informazioni utili ad un prossimo caricamento

Service lifecycle multithreading

Viene creato un Thread per ogni richiesta. Il thread può essere riutilizzato se lo stesso client richiede la servlet.