



# Spider

Armitano Joshua, Ballario Marco,  
Garro Christian, Gastaldi Paolo,  
Girauda Roberto, Mullace Matteo,  
Olivero Emanuele

# Obiettivo

Realizzare uno spider che:

- conta le hit di determinati professori su internet
- realizza una classifica dei professori più citati sul web

# Tecnologie utilizzate

- LAMPP
- Scrapy
- Bootstrap

# Parti principali

- Interfaccia utente
- Lancio degli spider
- Ricerca
- Database

# Database

indirizzi	
💡 idUrl INT	
◇ dominio VARCHAR(128)	
◇ urlPartenza VARCHAR(256)	
Indexes	
PRIMARY	

professori	
💡 idProf INT	
◇ nomeCompleto VARCHAR(50)	
◇ contatore INT	
◇ selezionato BOOLEAN	
Indexes	
PRIMARY	

# GestioneDatabase.py

Classe Python manager del database.

Permette il dialogo in lettura e scrittura con la base di dati del progetto.

## ManagerDB

```
- dbName: String
- host: String
- user: String
- password: String
- connection: Object
- cursor: Object
- urlIndex: int
- cognomeIndex: String

+ ManagerDB(dbName, host, user, password)
+ connetti(): void
+ chiudi(): void
+ aggiungiNome(nome): void
+ getElencoNomi(): String[]
+ aggiungiUrl(url): void
+ eliminaUrl(url): void
+ classificaNomi(): String[]
+ classificaHit(): String[]
+ leggiUrl(): String[]
+ leggiNome(): String
```

# aggiungiNome(self, nomi)

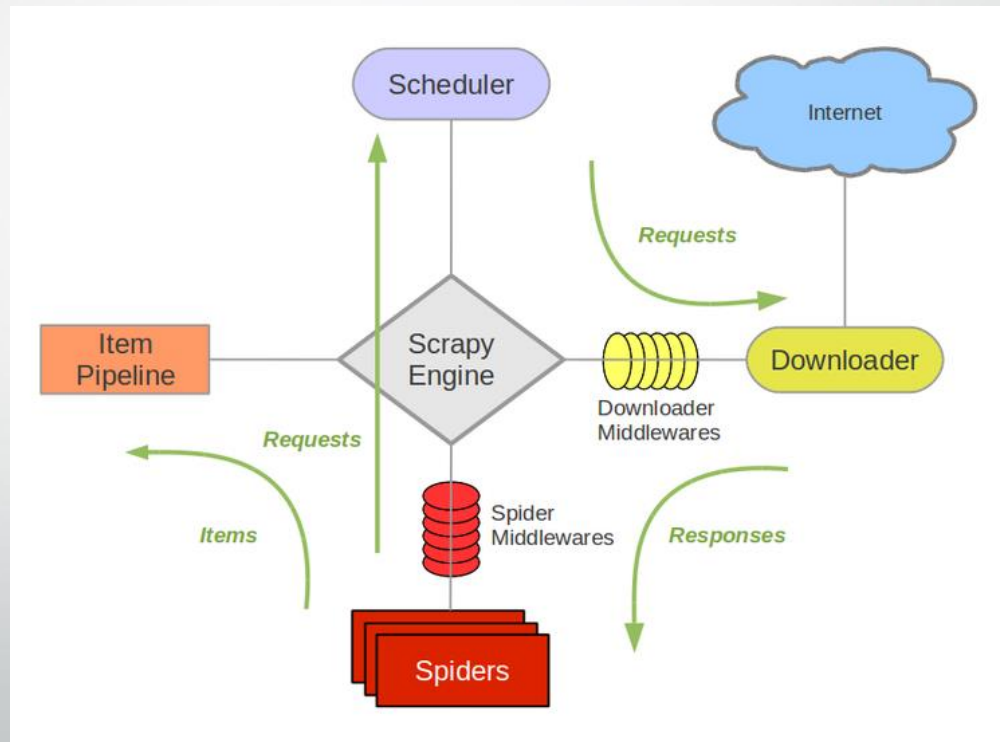
```
# Metodo per l'aggiunta di un risultato al db. Controlla la presenza di un nome e in caso
# affermativo ne va a incrementare il contatore. Gestisce anche array di nomi completi.
def aggiungiNome(self, nomi):
    try:
        self.__cursor = self.__connessione.cursor()
        self.__cursor.execute('SELECT nomeCompleto FROM professori ORDER BY nomeCompleto') #ricava tutti i nomi completi
        items = self.__cursor.fetchall()

        for singoloNome in nomi:
            for item in items: #scorre tutti i nomi completi in cerca di riscontri
                if item[0].lower() == singoloNome.lower():
                    self.__cursor.execute("UPDATE professori SET contatore = contatore + 1 WHERE nomeCompleto='%s'" % (singoloNome))
            self.__connessione.commit()

    except MySQLdb.Error, e:
        print "Error %d: %s" % (e.args[0],e.args[1])
        #exit(3)
```

# Scrapy

Scrapy è un framework Python di web crawling libero e open source. È utilizzato per il web scraping, per estrarre dati utilizzando le API o come web crawler.





# XPath

XPath è un linguaggio, parte della famiglia XML, che permette di individuare i nodi all'interno di un documento XML.

Le espressioni XPath, a differenza delle espressioni XML, non identificano la struttura di un documento, ma ne localizzano con precisione i nodi.

# MySpider.py

Classe fondamentale con il compito di:

- Scaricare pagine web
- Estrarre contenuti da elaborare

## MySpider

- name: String
- allowed\_domains: String[]
- start\_urls: String[]

- + MySpider(allowed\_domains, start\_urls)
- + parse(rsponse)

# MySpider.py

```
from scrapy.spiders import Spider
from scrapy.selector import Selector
from items import ElementoDaEsaminare

class MySpider(Spider):
    name = "spiderRicerca"
    allowed_domains = []
    start_urls = []

    def __init__(self, allowed_domain, start_url):
        self.allowed_domains.append(allowed_domain)
        self.start_urls.append(start_url)

        print "Spider> analizzando " + allowed_domain + "..."

    def parse(self, response):
        analizzatorePagina = Selector(response)
        paginaSito = ElementoDaEsaminare()
        paginaSito['body'] = analizzatorePagina.xpath('//body//text()').extract() #estrae tutto il corpo della pagina

        print "Spider> pagine ricevute"
        return paginaSito
```

# pipelines.py

```
class SpiderPipeline(object):
    def process_item(self, item, spider):
        gestoreDatabase = ManagerDB() #vedi altra feature
        gestoreDatabase.connetti()

        print "Spider> processando le pagine..."

        elencoProfessori = gestoreDatabase.getElencoCognomi()

        for prof in elencoProfessori:
            print "Spider> ricercando " + prof

        #item['body'] = [element.lower() for element in item['body']] #tutti gli elementi in minuscolo
        corpoPagina = ''.join(item['body']) #dalla lista ottengo un array
        corpoPagina.lower() #stringa tutta in minuscolo

        for professore in elencoProfessori:
            if corpoPagina.find(professore.lower()) >= 0 : #find() restituisce l'indice dell'occorrenza nella stringa):

                print "Spider> trovato " + professore
                gestoreDatabase.aggiungiCognome([professore]) #vedi altra feature
        #raise DropItem("Elemento non piu' utile") #eliminazione dell'item (non obbligatoria) (lancia un'eccezione, pertanto non e' stata abilitata)

        gestoreDatabase.chiudi()
```

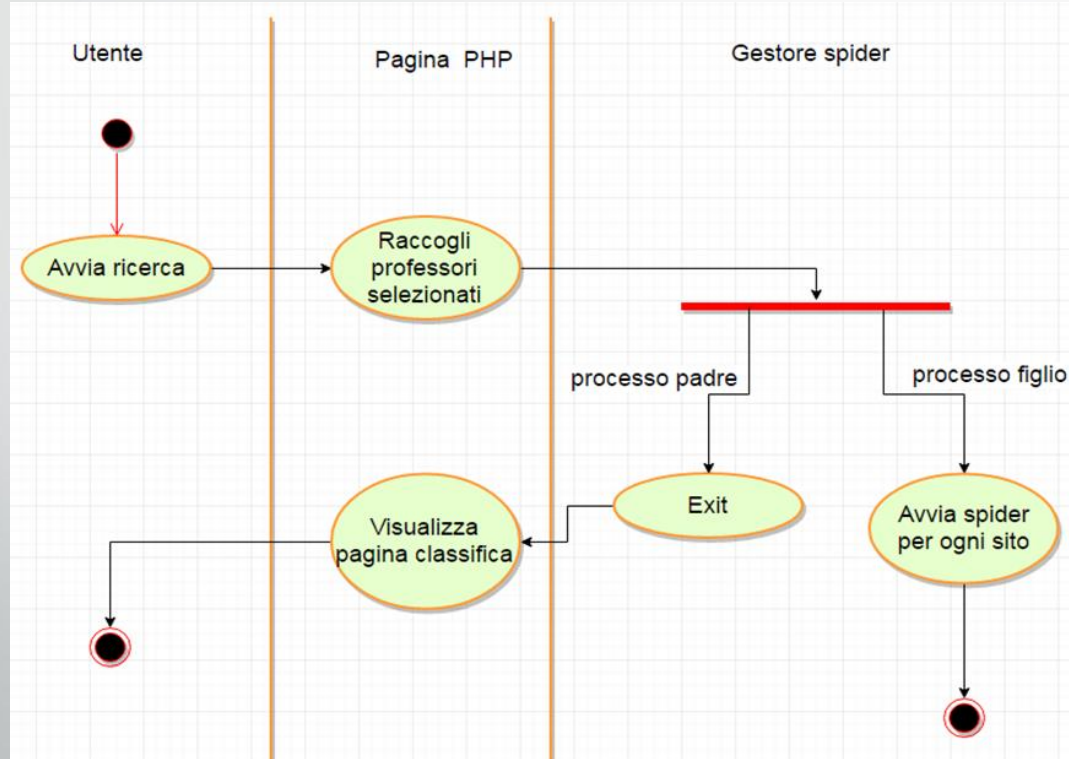
# Avvio degli spider

Fasi da eseguire:

1. Tenere traccia dei professori selezionati
2. Aggiornare il database
3. Avviare il gestore degli spider
4. Visualizzare la classifica

# Avvio degli spider

-diagramma soluzione-



Creazione di un processo zombie

# avviaSpider.php

```
/*connessione*/
$dbh = new PDO("mysql:host=$hostname;dbname=$database_name", $username, $password);

/*deselezione di tutti i campi*/
$stmt = $dbh->prepare('UPDATE professori SET selezionato=FALSE, contatore=0'); /*stmt = statement*/
$stmt->execute();

/*seleziono solo i prof che sono stati selezionati dell'utente*/
foreach($_GET['elencoProfSelezionati'] as $professore){ //elencoProfSelezionati è un array contenente le checkbox selezionate
    $stmt->closeCursor();
    $stmt = $dbh->prepare('UPDATE professori SET selezionato=TRUE WHERE idProf=:idProf'); //stmt = statement
    $stmt->bindParam(':idProf', $professore, PDO::PARAM_INT);
    $stmt->execute();
}

$cmd = escapeshellcmd('python ' . __DIR__ . '/gestoreSpider.py');
echo "cmd: $cmd <br>";
echo shell_exec($cmd);

/*disconnessione*/
$dbh = null;

/*redirecting alla pagina dei risultati*/
header("Location: classifica.php");
die();
```

# gestoreSpider.py

```
#questo crea un processo zombie, ma almeno il
#processo chiamante (la pagina php) può andare avanti
if os.fork() > 0:
    print "Ricerca avviata"
    sys.exit(0)
```

```
#creazione del gestore del database
managerDB = ManagerDB()
managerDB.connetti()
```

```
#lista dei siti su cui avviare gli spider
websites = managerDB.leggiUrl()
```

```
# crawlers in esecuzione
crawlers = []
crawlersCompletati = 0;
```

```
#funzione per gestire l'evento sulla chiusura dello spider
def spider_closing(spider):
    crawlersCompletati = crawlersCompletati+1

    #se hanno completato tutti i gli esecutori degli spider chiudo il programma
    if crawlersCompletati == len(websites):
        print "finito"
        bloccaProcesso.stop()

#per ogni sito lancio uno spider tramite un suo esecutore
for i in range(len(websites)):
    # crawl responsibly
    #settings.set("USER_AGENT", "Kiran Koduru (+http://kirankoduru.github.io)")

    #imposto le impostazioni
    setting = get_project_settings()
    setting.setmodule(settings, 300)

    #creo un nuovo esecutore di uno spider
    crawler = Crawler(MySpider, setting)

    crawlers.append(crawler)

    #aggiungo all'esecutore dello spider una funzione sull'evento di chiusura dello spider
    crawler.signals.connect(spider_closing, signal=signals.spider_closed)

    #avvio dello spider passando gli appositi parametri
    crawler.crawl(websites[i][0], websites[i][1])

#per bloccare il processo, altrimenti terminerebbe prima dei crawler
bloccaProcesso.run()
```





# Interfaccia utente

Tre pagine principali:

1. [index.php](#)
2. [classifica.php](#)
3. [link.php](#)

# Index.php

## Progetto: Spider

[Ricerca](#)[Classifica](#)[Link](#)

Ricerca

Cognome Professore	Conteggio Professore	Aggiungi alla ricerca	Azioni	
<input type="text" value="Tosello Giovanni"/>	0	<input type="checkbox"/>	<button>Salva</button>	<button>Elimina</button>
<input type="text" value="Pasquale Alessandra"/>	0	<input type="checkbox"/>	<button>Salva</button>	<button>Elimina</button>
<input type="text" value="Rosso Maurizio"/>	0	<input type="checkbox"/>	<button>Salva</button>	<button>Elimina</button>
<input type="text" value="Girauda Giuseppe"/>	0	<input type="checkbox"/>	<button>Salva</button>	<button>Elimina</button>
<input type="text"/>			<button>Aggiungi</button>	

Avvia ricerca

Progetto spider di: Armitano Joshua, Ballario Marco, Garro Christian, Gastaldi Paolo, Girauda Roberto, Mullace Matteo e Olivero Emanuele.

# Classifica.php



Progetto: Spider

Classifica

[Ricerca](#)

[Classifica](#)

Cognome professore	Conteggio
Rosso Maurizio	0
ITIS	0
Ricerca	0
Sperotto Sergio	0
Girardi Stefano	0
Mina Guido	0
Rosa Guido	0
Puppo Marco	0

[Aggiorna](#)

# Link.php

## Progetto: Spider

[Ricerca](#)[Classifica](#)[Link](#)

Ricerca

### Aggiungi Indirizzi

Aggiungi

Progetto spider di: Armitano Joshua, Ballario Marco, Garro Christian, Gastaldi Paolo, Giraudo Roberto, Mullace Matteo e Olivero Emanuele.

# Repository ufficiale

<https://github.com/Onhyro/spider.git>