SEP	17
Title	Spider Contracts
Author	Insophia Team
Created	2010-06-10
Status	Draft

SEP-017: Spider Contracts

The motivation for Spider Contracts is to build a lightweight mechanism for testing your spiders, and be able to run the tests quickly without having to wait for all the spider to run. It's partially based on the [https://en.wikipedia.org/wiki/Design_by_contract Design by contract] approach (hence its name) where you define certain conditions that spider callbacks must met, and you give example testing pages.

How it works

In the docstring of your spider callbacks, you write certain tags that define the spider contract. For example, the URL of a sample page for that callback, and what you expect to scrape from it.

Then you can run a command to check that the spider contracts are met.

Contract examples

gExample URL for simple callback

The `parse_product` callback must return items containing the fields given in `@scrapes`.

```
#!python
class ProductSpider(BaseSpider):

    def parse_product(self, response):
        """
        @url http://www.example.com/store/product.php?id=123
        @scrapes name, price, description
        """
```

gChained callbacks

The following spider contains two callbacks, one for login to a site, and the other for scraping user profile info.

The contracts assert that the first callback returns a Request and the second one scrape `user, name, email` fields.

```
#!python
class UserProfileSpider(BaseSpider):

    def parse_login_page(self, response):
        """
        @url http://www.example.com/login.php
        @returns_request
        """
        # returns Request with callback=self.parse_profile_page

    def parse_profile_page(self, response):
        """
        @after parse_login_page
        @scrapes user, name, email
        """
        # ...
```

Tags reference

Note that tags can also be extended by users, meaning that you can have your own custom contract tags in your Scrapy project.

`@url`	url of a sample page parsed by the callback
`@after`	the callback is called with the response generated by the specified callback
`@scrapes`	list of fields that must be present in the item(s) scraped by the callback
`@returns_request`	the callback must return one (and only one) Request

Some tag constraints:

- a callback cannot contain `@url` and `@after`

Checking spider contracts

To check the contracts of a single spider:

```
scrapy-ctl.py check example.com
```

Or to check all spiders:

```
scrapy-ctl.py check
```

No need to wait for the whole spider to run.