Crawler 구현 문서

Twitter 및 Naver & Daum Blog를 crawling 한다.

1. DB 연결

mongod --dbpath D:\Project\ITSMINE\mongoDB\ --logpath D:\Project\ITSMINE\mongoDB\log\log.txt –directoryperdb

1. API 사용
   1. Naver
      1. 요청 URL

http://openapi.naver.com/search

* + 1. 요청 변수

|  |  |  |
| --- | --- | --- |
| 변수 | 값 | 설명 |
| Key | String | Api 키값(필수) |
| Target | String | 필수, Blog(블로그), webkr(웹문서),  cafearticle (카페) |
| Query | String | 질의, UTF-8 기본 |
| Display | Int | 검색 출력 건수 10~100 |
| Start | Int | 시작위치 1~1000 |
| Sort | String | Sim(유사도순), date(날짜순) |
| Api key | a4c87ac3748a15f672bb5f644443115f | |

* + 1. 응답 결과
    2. Python code

|  |
| --- |
| \_\_author\_\_ = 'KCP' \_\_date\_\_ = '2015-05-06'  **from** urllib **import** parse, request **import** xml.etree.ElementTree **as** et   **def getsearchresult**(target, query):  *"""Return a xml tree or xml root element  Keyword argument:  target -- Request URL  query -- Dictionary for search  """* xmldata = request.urlopen(target+'?'+parse.urlencode(query)).read().decode('utf-8')  xmltree = et.fromstring(xmldata)  **return** xmltree  a = getsearchresult('http://openapi.naver.com/search', {'key': 'a4c87ac3748a15f672bb5f644443115f', 'target': 'blog', 'query': '제주도', 'display': 10, 'start': 1, 'sort': 'sim'}) |

* + 1. 특이사항
  1. Daum
     1. 요청 URL
     2. 요청 변수
     3. 응답 결과
     4. Python code
     5. 특이사항
  2. Twitter
     1. 요청 URL
     2. 요청 변수
     3. 응답 결과
     4. Python code
     5. 특이사항
  3. Tour API
     1. 요청 URL
     2. 요청 변수
     3. 응답 결과
     4. Python code
     5. 특이사항