**개발 코드 및 개발 문서**

20150224 김우석

**사용 기술**

파이썬 3.7.9

파이썬 프레임워크 scrapy

AWS lambda

AWS dynamoDB

AWS EC2, 카카오 I 오픈빌더

전체 구조를 요약하면 파이썬 프레임워크 scrapy를 사용하여 숭실대학교 기숙사 식단 페이지와 숭실대학교 생활협동조합 주간 식단 페이지에서 학식 메뉴를 일주일 단위로 크롤링하여 AWS dynamoDB에 저장합니다. 일주일 중 월요일에 식단이 업데이트되기에 EC2와 cron을 사용하여 매주 월요일 03시에 scrapy가 실행되도록 했습니다.

사용자가 카카오톡 챗봇을 통해 특정 발화를 입력하면 POST 메서드로 RESTful API가 호출되어 매핑된 AWS lambda 함수가 실행됩니다. 함수의 인자로 들어오는 발화에 맞는 정보를 dynamoDB로부터 가져온 뒤 JSON으로 구성하여 반환하면 챗봇이 사용자에게 전달합니다.

주요 API로는 scrapy를 이용한 웹 스크래핑 api, 사용자가 호출하는 챗봇 api, 챗봇을 통해 호출되는 AWS lambda api가 있습니다.

**목차**

- scrapy

- 카카오 챗봇

- AWS lambda



**1. scrapy**

본 프로젝트에서 사용한 scrapy의 구성은 다음과 같습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

중점으로 볼 것은 items.py, pipelines,py, spiders 디렉터리 안의 myspider.py입니다.

**Item**

우선 items.py입니다. item은 스크래핑한 데이터를 저장하는 key-value 형식의 자료구조입니다.

해당 프로젝트에서는 기숙사 식단을 저장하는 Dormitory와 학생식당 식단을 저장하는 Student를 item으로 사용했습니다.



저장하는 데이터는 날짜를 나타내는 date와 조식, 중식, 석식 중 어느 것에 해당하는지 나타내는 time, 장소를 나타내는 location, menu로 구성되어 있습니다.

**Spider**

spider는 코드의 양이 길어 핵심 부분만 설명하겠습니다. 앞서 말했듯이 spider는 웹페이지를 스크래핑하는 클래스입니다.



name은 spider를 구분하기 위한 이름이고, allowed\_domains는 spider가 링크를 따라 스크래핑하다가 다른 도메인에 접근하는 것을 방지하기 위한 것입니다. 본 프로젝트에서는 학생식당 웹사이트와 기숙사식당 웹사이트에 접근합니다.

기숙사식당의 경우 현재 날짜를 URL의 쿼리스트링으로 구성하는데, 토요일이나 일요일일 경우 다음 주 식단이 보이기 때문에 금요일이 되게 설정했습니다. 그리고 학생식당의 경우 이번 주를 기준으로 j값을 이용해 저번 주는 -1, 다음 주는 1과 같은 형식으로 해당 주의 식단을 보여줍니다. 일요일을 기준으로 하고 있기 때문에 일요일일 경우 j를 -1로 구성하여 이번 주의 메뉴를 보여줄 수 있도록 설정했습니다.

start\_urls는 spider에게 시작 URL을 지정해주는 것으로, spider는 해당 URL을 scrapy engine에게 request로 보내고 여러 과정을 거쳐 페이지의 정보가 담긴 response를 받아옵니다.

start\_urls를 기점으로 spider는 웹페이지를 스크래핑하며, logic에 따라 해당 웹페이지에 있는 다른 URL을 타고 또 다른 request를 구성하여 engine에게 전달해줄 수도 있습니다.



각 request에 대해 response를 처리하기 위해 호출되는 메서드입니다. response 에는 downloader가 fetch한 페이지의 정보가 들어있습니다. 기숙사 식당의 주소인지 비교하고, 맞으면 parse\_dormitoy 메서드를 실행하고 아니면 parse\_student 메서드를 실행합니다.



기숙사 식당 사이트의 정보를 스크래핑하는 메서드입니다. 식단 정보는 td에 있는데, <string>이나 <b>같은 태그들이 text를 감싸고 있는 경우도 있어 xpath(‘string()’)으로 추출해줍니다. item을 구성할 변수들을 선언하고, 중.석식이라는 공백란은 루프에서 제외합니다.

식단 정보를 다듬은 뒤 item을 구성하고, yield를 사용하여 반복적으로 return합니다. 이렇게 되면 parse에서도 반복적으로 yield를 통해 item을 pipeline으로 보냅니다.



학생 식당 페이지의 식단은 태그가 일정하지 않아 점심 3코너가 있을 때와 없을 때로 구분했습니다. 학생 식당의 식단 정보는 설명글이 붙어있기 때문에 전체가 아닌 메인 메뉴만 스크래핑하여 item을 구성합니다. extractMenu 메서드를 호출하여 메인 메뉴를 스크래핑합니다.



메인 메뉴 다음에 영어로 메인 메뉴를 번역한 문자열이 오는데, 태그가 일정하지 않기 때문에 이 정보를 활용하여 메인 메뉴를 스크래핑합니다. 만약 div태그가 2개 미만이라면 그 날은 메뉴가 없는 날이기 때문에 빈 문자열을 반환합니다.

**Pipeline**

다음은 pipeline입니다. spider가 시작되거나 종료될 때, item이 생성될 때 같은 특정 상황에서 실행할 logic을 정해줄 수 있습니다.



\_\_init\_\_은 pipeline이 생성될 때 최초 1회 실행됩니다. AWS의 dynamodb와 연결을 시도하고, 실패할 경우 종료합니다.

spider\_opened는 spider가 실행될 때 실행됩니다. dynamodb와의 연결이 성공했을 경우에만 실행됩니다.

process\_item은 item이 리턴될 때마다 실행됩니다. dynamodb의 테이블이 id, type, location, menu로 구성되어 있는데 id에 중복되지 않은 값을 넣기 위해 코드를 구성한 뒤 dynamodb에 데이터를 넣습니다.

spider\_closed는 spider가 종료될 때 실행됩니다.

위와 같은 logic으로 웹페이지로부터 식단 정보를 스크래핑하여 AWS의 dynamodb에 저장하면 scrapy의 역할은 끝이 납니다.

클립아트이(가) 표시된 사진

자동 생성된 설명

**2. 카카오톡 챗봇**

본 프로젝트에서 사용한 발화 패턴은 11개로, 다음과 같습니다.

- 오늘, 내일, 월, 화, 수, 목, 금, 토, 일. 처음으로, 요일별

사용자가 해당 발화 패턴에 없는 발화를 입력했을 경우 챗봇은 ‘풀백 블록’을 실행하여 이해하지 못했다는 의사를 표현합니다.

위의 발화 패턴이 들어오면 챗봇은 설정된 ‘스킬’을 실행합니다. ‘스킬’은 정해진 응답을 출력하는 것이 아니라, 스킬 서버로 요청을 보내고 적절한 응답을 반환 받아 출력하는 것입니다.

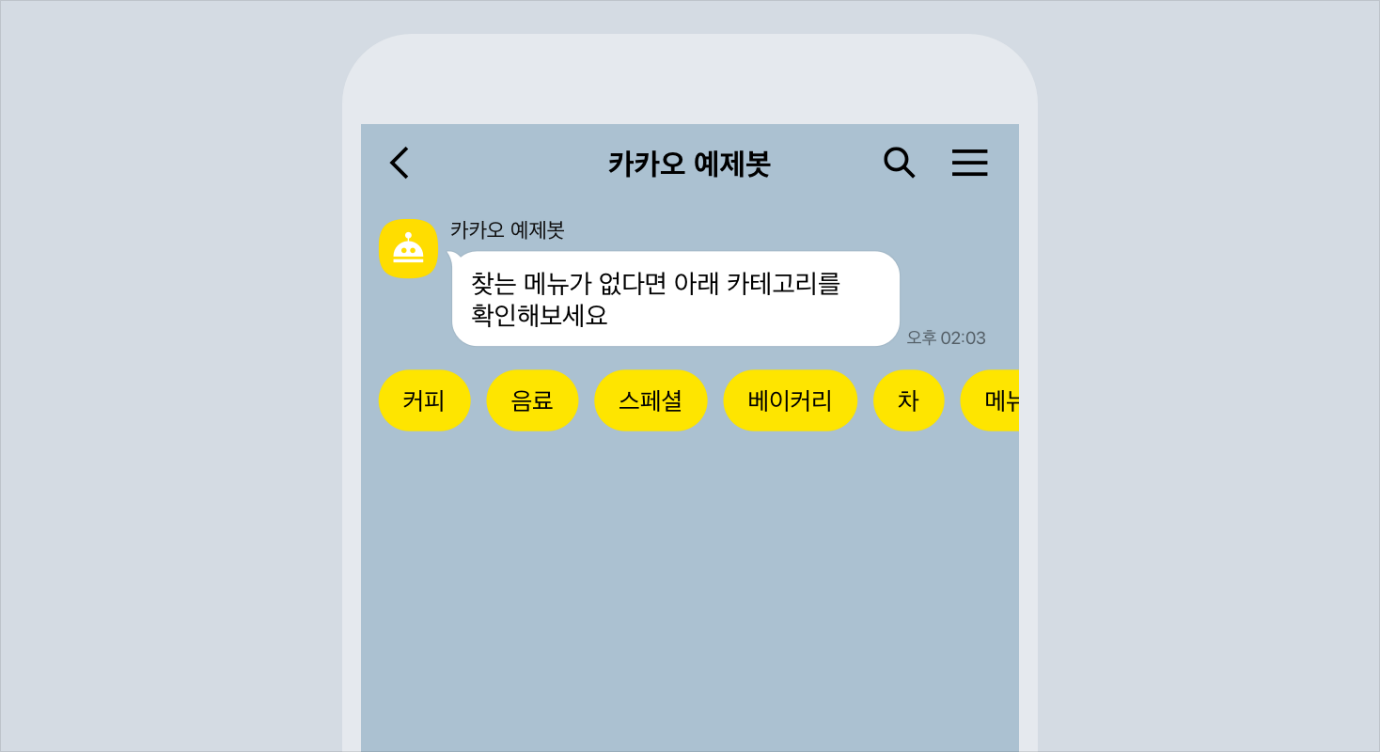
AWS lambda를 실행하는 REST API를 POST로 호출합니다. request body는 다음처럼 구성됩니다.



여기서 사용하는 정보는 userRequest의 utterance입니다. 사용자가 입력한 발화가 들어갑니다.

챗봇은 해당 request를 ‘https://voc6zpb3z0.execute-api.ap-northeast-2.amazonaws.com/test/ssubobbot’로 보냅니다. 그러면 AWS lambda가 실행되고, lambda로부터 JSON으로 된 response를 받습니다.

이 response로 카카오톡 챗봇의 응답 형태가 정해집니다. 그렇기 때문에 다양한 형태가 있는데 본 프로젝트에서는 SimpleText와 QuickReplies 형태로 챗봇이 응답합니다. SimpleText는 단순히 문자열만 전송하는 형식이고, QuickReplies는 사용자가 누르면 매핑된 대화가 입력되기에 사용하기 편해지는 형식이라고 할 수 있습니다.



SimpleText + QuickReplies는 위와 같은 응답을 말합니다.

모든 응답의 기본 JSON 형식은 다음과 같습니다.



version은 스킬 응답의 버전으로 **<major-version>.<minor-version>**의 모습을 갖습니다. version이 없다면 구 버전의 응답으로 간주하기 때문에 항상 포함해야 합니다.

template은 스킬 응답의 출력 모양을 담고 있는 항목입니다.

context는 블록의 context 설정을 제어할 수 있습니다.

data는 필요에 따라 커스텀한 데이터를 넣을 수 있는 항목입니다. 본 프로젝트에서는 사용하지 않습니다.

챗봇은 응답으로 온 JSON을 파싱하여 출력하는 역할이기 때문에 프로젝트에서 쓰인 JSON 형식은 AWS lambda에서 설명하고 지금은 형식만 설명하겠습니다.

SimpleText의 형식은 다음과 같습니다.



“simpleText”의 키로 “text”키와 value가 옵니다. 여러 개가 올 수 있습니다.

QuickReplies의 형식은 다음과 같습니다.



“label”이 표시되는 문자열이며, “action”은 message 혹은 block을 정할 수 있습니다. message는 “message”의 데이터를 그대로 입력하며, block은 blockId를 따로 지정하여 입력되는 데이터와 관계없이 정해진 블록을 호출합니다. “message”는 선택 시 입력되는 문자열입니다.



**3. AWS lambda**

본 프로젝트에서 사용한 lambda 함수는 API 게이트웨이와 연결되어 있어 REST API를 호출할 때 실행되도록 설계했습니다. 전체 코드는 다음과 같습니다.



lambda의 main 메서드입니다. event는 request 객체이고, context는 response 객체입니다. event는 요청에 대한 정보들이 들어있으며, context에는 done(), succeed(), fail()이 있어 각각의 상황에 전달할 response를 구성할 수 있습니다.

본 프로젝트에서는 카카오톡 챗봇에 의해 lambda가 실행되므로, request에서 ‘utterance’에 저장된 value를 추출합니다.

lambda가 호출되면 우선 챗봇의 quickReplies를 구성합니다.

사용자가 ‘처음으로’나 ‘요일별’을 발화로 입력했을 경우 dynamoDB에서 정보를 가져오지 않고 추가적인 정보를 요구하는 JSON을 반환합니다. ‘처음으로’의 경우 ‘오늘’, ‘내일’, ‘요일별’이라는 quickReplies와 함께 입력하라는 simpleText를 구성하여 반환합니다. ‘요일별’의 경우 요일에 해당하는 quickReplies와 함께 입력하라는 simpleText를 구성하여 반환합니다.

그 외의 경우 get\_date 메서드를 호출하여 발화에 담긴 날짜 정보를 연산합니다. ‘오늘’의 경우 오늘 날짜를 연산하고 ‘내일’은 내일의 날짜를 연산합니다. 요일이 입력됐을 경우도 같습니다.

그 후 get\_menu 메서드를 호출하여 dynamoDB에서 해당 날짜의 학식 정보를 가져와서 JSON을 구성하여 반환합니다. 일요일에 내일 정보를 물어봤을 경우, 웹사이트에 학식 정보 업로드가 평균적으로 일요일 저녁 10시에 업로드되기 때문에 메뉴가 없다는 simpleText를 구성하여 반환합니다.

메뉴가 있을 경우 메뉴를 simpleText로 보기 좋게 정렬하여 반환합니다.



get\_date 메서드는 사용자가 날짜 정보가 담긴 발화를 input으로 받아 실행되는 메서드입니다. 해당 날짜를 문자열로 연산해서 ‘YYYYMMDD’ 형식으로 output으로 반환합니다.



get\_menu 메서드는 날짜를 input으로 받아 그 날의 학식 정보를 dynamoDB로부터 읽어와 반환하는 메서드입니다. ssubobbot이라는 테이블로부터 id가 date로 시작하는 항목들을 가져옵니다. DB와 상호작용에 실패할 경우 빈 dict 객체를 반환합니다.



set\_simpleText 메서드는 사용자가 입력한 발화가 ‘처음으로’, ‘요일별’이거나 일요일에 ‘내일’을 입력했을 경우 실행되는 메서드입니다. input으로 발화를 받습니다. 전자의 경우 다음 정보를 요구하는 simpleText를 구성하여 output으로 반환하고, 후자의 경우 해당 날짜의 메뉴가 없다고 말하는 simpleText를 반환합니다.



set\_menu\_simpleText 메서드는 메뉴까지 성공적으로 가져온 뒤 실행되는 메서드입니다. 날짜와 메뉴를 input으로 받아 보기 좋게 정리하여 simpleText로 반환합니다.



set\_quickReplies는 발화를 input으로 받아 quickReplies를 구성하여 반환하는 메서드입니다. ‘label’, ‘action’, ‘messageText’에 해당하는 value를 저장합니다.

사용자가 ‘처음으로’를 입력한 경우 ‘오늘’, ‘내일’, ‘요일별’ 블록을 quickReplies로 반환합니다.

사용자가 ‘요일별’을 입력한 경우 각 요일과 ‘처음으로’ 블록을 quickReplies로 반환합니다.

그 외의 정보는 정의하지 않은 발화들이므로 ‘처음으로’ 블록만 quickReplies로 구성하여 반환합니다.

lambda는 위와 같은 행위를 한 후 simpleText와 quickReplies를 챗봇이 알아들을 수 있는 JSON으로 구성하여 반환합니다.