* **목적**

네이버 공연정보 data를 크롤링하여 db에 저장 후, 저장된 데이터로 장르별 공연정보를 화면으로 제공한다.

* 데이터 : 네이버공연정보 > Network에서 공연정보가 제공되는 페이지를 이용한다.

(페이징되어 제공되는 공연정보가 파라미터로 보내지지 않아도, network로 전달되는 url로 확인하여 내용을 크롤링 할 수 있다.)

* 크롤링 & DB 저장 : /crawling에서 제공하는 버튼클릭 시, "금일 날짜"기준의 공연정보크롤링
* 크롤링 수행 & db 저장 : <http://localhost:8080/crawling>
* 연극/뮤지컬 카테고리 선택 (category, genre) /
* 장르선택 : 장르에 해당하는 공연정보 (title, thum, location, period)확인 가능
* 상세페이지 : [예매하기] 버튼으로 공연예매페이지로 이동, 관련 youtube 동영상 정보 제공
* 네비게이션 : 1)~3)페이지로 이동
* 찜하기 기능 : 원하는 공연을 따로 저장할 수 있다.
* 로그인 기능

* **정보**
* 크롤링 : Jsoup이용
* DB : Mysql databede, JPA 사용
* Mysqul : 쿼리로 genre, category 선택해서 그에 해당하는 data를 가져와서 뿌려줄 수 있다.

사용 : like검색이용 (%genre%)로 검색 용이, 데이터 I/O를 빠르게 수행할 수 있다.

* \*\*Jpa 특징 : 인터페이스, 모델 클래스에 테이블 정보 작성하면 자동으로 생성, 메소드명으로 쿼리를 보낼 수 있다.

인터페이스(ContentRepository.java) : 쿼리 작성, jpaRepository 임플리먼츠

모델클래스(ContentInfo.java) : 클래스명-table 명, 테이블 필드 작성

table<"content\_info"> 자동생성(필드, 자료 Type 등 jpa가 생성)

* **기능**

3.1 데이터 크롤링 : ContentInfo Controller.java

* Joup 라이브러리 사용(parse, select로 태그, 클래스, attr선택), 네이버 공연페이지
* 데이터 정제 : replaceAll로 select할 대상을 정제한 후, 태그로 선택하여 필요한 정보를 parsing한다.
* 데이터(6개) : **공연정보의 이름(title), 썸네일(thumb), 상영기간(period), 장르(6가지), 장소(location), category(연극/뮤지컬)**
* 카테고리, 장르에서 가져온 데이터들을 db에 저장한다.

3.2 데이터 db 저장

* 컨트롤러에서 db 저장 로직 작성 : ContentInfo Controller.java
* 이미 존재하는 데이터인지 자료의 갯수로 확인한다.

int existcount = contentRepository.existcount(title, category);

기준 : title, category

* 1)에서 이미 존재(개수>0)하면 "장르(genre)"는 ","로 구분하여 추가하고, 나머지 데이터는 다시 업데이트 한다
* **장르별로 컬럼을 생성해서 관리하지 않고 크롤링 할 때마다 장르 부분만 추가해서 update를 한다.**

=> like 검색으로 해당 기능 구현이 가능하므로 구분자(,) 로 구분했다.

* 'genre' 컬럼은, 값만 추출하면 되는 기능으로 사용하며 genre가 늘어날 수 있는 상황을 고려할 때 컬럼생성을 불필요하게 하지 않을 수 있다.
* 이미 존재하는 데이터인지 비교 : Title, category의 값이 이미 저장되어있으면, "장르(genre)" 만 넣는다.
* 1)에서 자료가 존재하지 않으면, data를 save한다.

3.2.2 인터페이스에서 db명령 ContentRepository.java

* 쿼리 : select, update, save, return List<ContentInfo>
* 장르추가 쿼리 : genre는 concat으로 ','로 장르를 추가

@Query("update content\_info set period=:period,thumb=:thumb,genre=concat(genre, ',', :genre) where title=:title and category=:category ")

void updatecontent(@Param("period") String period,@Param("thumb") String thumb,@Param("genre") String genre,@Param("title") String title,@Param("category") String category);

* 화면

3.3.1. /bigdoor 화면 : 연극 || 뮤지컬 (category) 선택 시 파라미터로 category, genre 제공

예. category=stage&genre=""

* 기본값 : genre="" (전체)로 제공되어, 연극/뮤지컬에 해당하는 "전체"리스트가 보여진다.

 

장르 선택 제공 : 장르<button>클릭 할 때마다 파라미터의 genre value가 변경되어 전달된다.

* 공연정보 제공
* 넘겨진 category, genre 선택값을 HomeController에서 받아, db에서 선택값으로 해당하는 Contents lists를 가져온다
* HomeController에서 db에서 가져온 contentLIst data들을 받아 stage.html을 return 한다.
* Html에서 타임리프를 사용해서 each 구문으로 data를 화면에 제공한다.

* 상세 페이지
* 컨텐츠 정보 부분 화면 구성
* stage.html에서 연극/뮤지컬 제목을 선택하면 해당 콘텐츠의 제목, 장소, 기간 등 정보를 detail.html에 파라미터로 전달
* 받은 파라미터를 화면에 뿌림
* 유튜브 동영상 가져오기
* ajax를 이용해 유튜브에 데이터를 요청, get으로 받아옴
* ajax에 쓰는 url 부분에 파라미터로 넘어온 제목을 합쳐서 데이터 요청
* 가져온 json 데이터를 파싱해 영상 화면에 출력
* 예약하기
* 예약하기 버튼을 누르면 해당 공연을 검색한 인터파크 티켓 화면으로 이동



* 찜하기
* 찜한 공연의 정보 전송
* stage.html 의 ‘찜하기’ 버튼을 누르면 몇 번째 것을 눌렀는지 확인
* 해당 번째의 연극/뮤지컬 정보를 ajax로 ‘/test’ 로 전달
* user 가 로그인 되었는지 확인
* ‘/test’ 에서 현재 로그인 한 user의 session을 받아옴
* session이 없으면 0 리턴, 있으면 1번에서 보낸 정보를 scrap\_db에 저장하고 1 리턴
* 로그인 여부에 따라 알람창 띄우기(찜했습니다 / 로그인이필요합니다 > 로그인창으로)



3. 찜한 공연

* 현재 로그인 한 user의 email을 받아서 scrap\_db의 email과 비교
* 일치하는 내용만 화면에 뿌려줌
* 예약하기 버튼을 누르면 해당 공연을 검색한 인터파크 티켓 화면으로 이동