

Story API

2020.6.3

서성윤

배경

- Google Assistant의 기능 중 하나
- 이야기를 랜덤하게 전달해주는 기능



재미있는 이야기해 줘



오락실을 지키는 용 두 마리는?

일인용과 이인용이지요 🙄

목적

- 다양한 종류의 이야기를 제공
 1. 제공하는 이야기 종류와 개수 확인하기
 2. 이야기 얻기
 3. 이야기 공유하기

기능 1

- 제공하는 이야기 종류와 개수 확인하기

1. GET + Path: /info

2. Request Body: 없음.

3. Response →

200 OK

```
{  
  "storyinfo": [  
    {  
      "category": "Funny",  
      "count": "1"  
    }  
  ]  
}
```

기능 2

- 이야기 얻기

1. GET + Path: /[카테고리 이름] e.g., /funny, /scary, /romantic

2. Request Body: 없음.

3. Response →


200 OK

```
{
  {
    "title": "몸이 너무 아파요",
    "content": "몸을 눌러보니까 모든 곳이 아팠어요. 알고보니 손가락이  
부러진거였어요 ^^;"
  }
}
```

기능 3

- 이야기 공유하기

1. Post + Path: /[카테고리 이름] e.g., /funny, /scary, /romantic

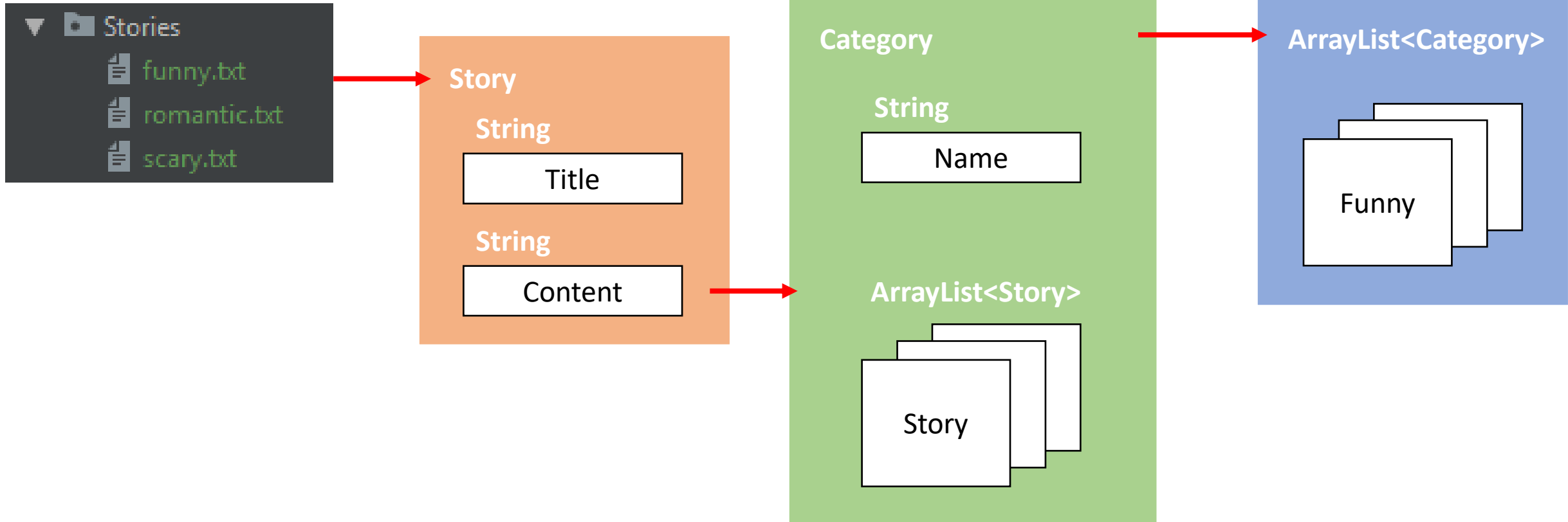
2. Request Body: 

```
{
  {
    "title": "몸이 너무 아파요",
    "content": "몸을 눌러보니까 모든 곳이 아팠어요. 알고보니 손가락이  
부러진거였어요 ^^;"
  }
}
```

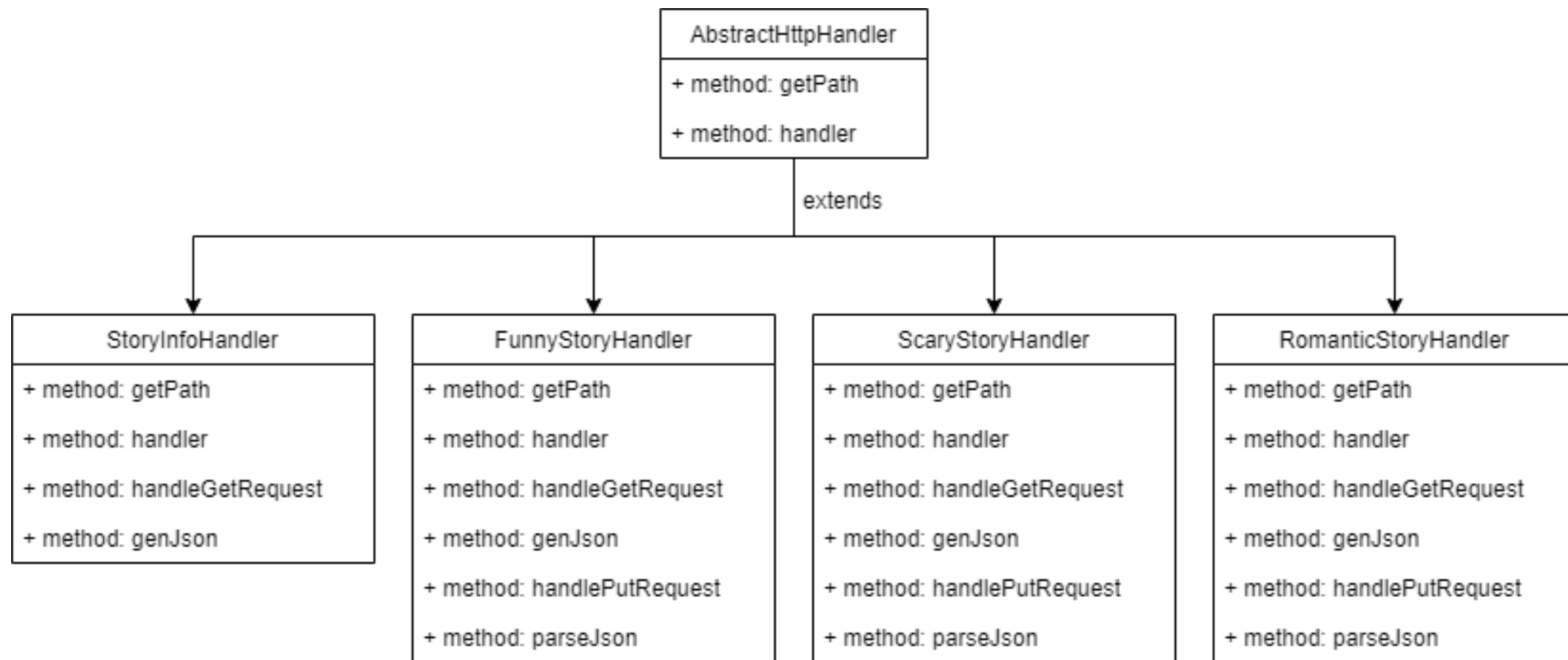
3. Response:

Status code: 200 OK, 400 Bad Request

구현 - 데이터

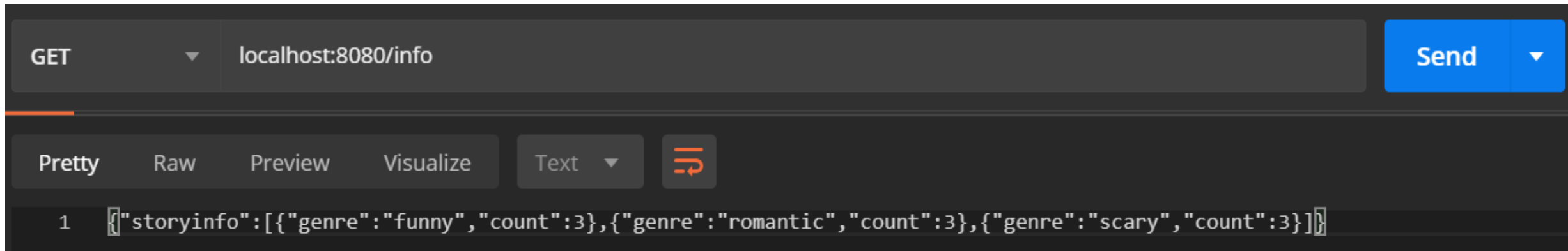


구현 - HTTP 서버



시연

- 제공하는 이야기 종류와 개수 확인하기

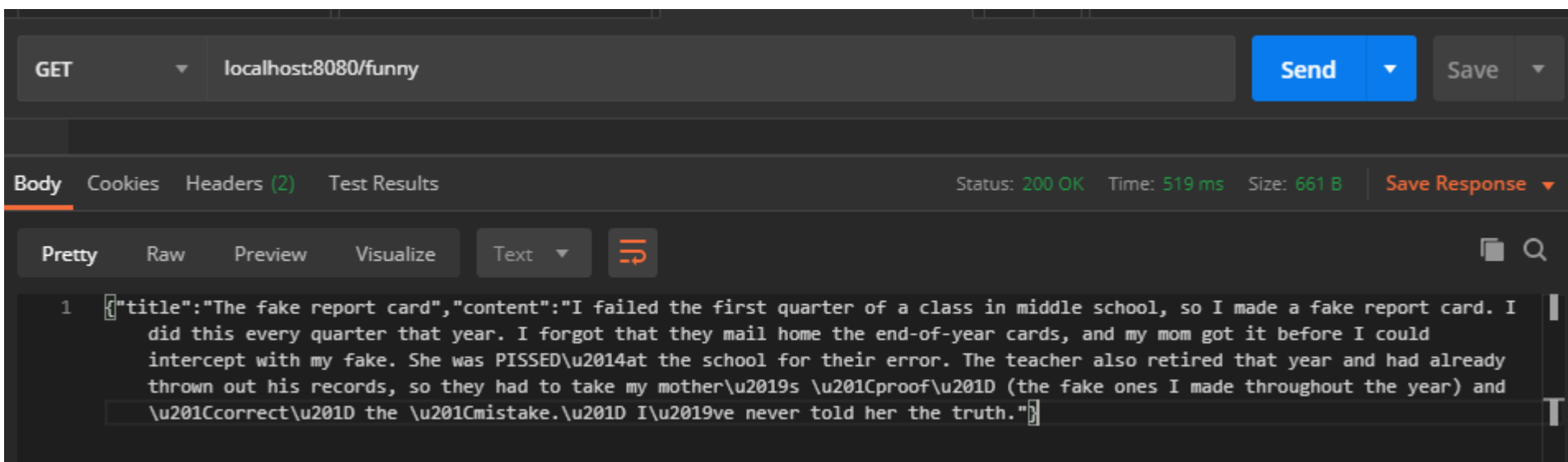


The screenshot shows a REST client interface with a dark theme. The top bar displays the HTTP method 'GET' and the URL 'localhost:8080/info'. A blue 'Send' button is on the right. Below the bar, there are tabs for 'Pretty', 'Raw', 'Preview', and 'Visualize', followed by a 'Text' dropdown and a refresh icon. The response area shows a JSON array with three objects, each representing a story genre and its count.

```
1 [{"storyinfo":[{"genre":"funny","count":3}, {"genre":"romantic","count":3}, {"genre":"scary","count":3}]}]
```

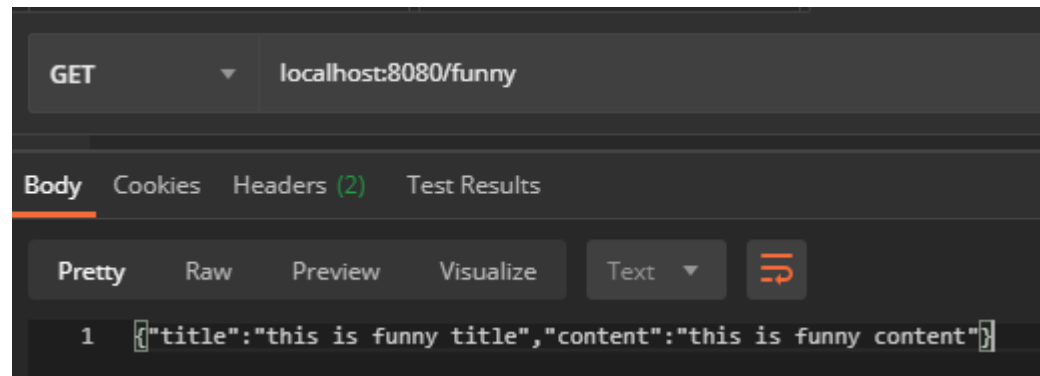
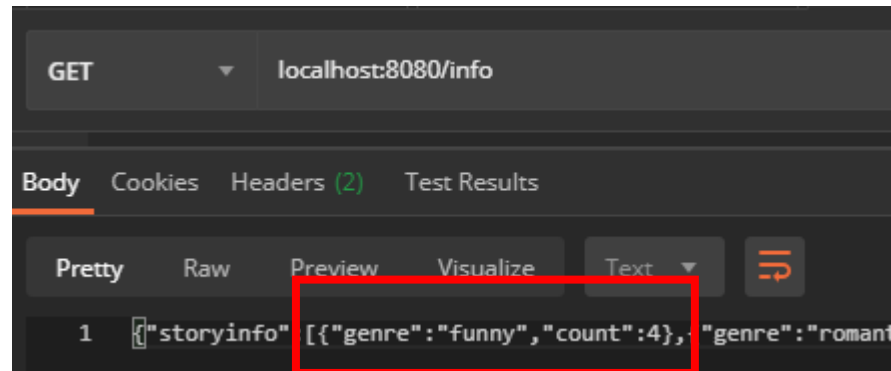
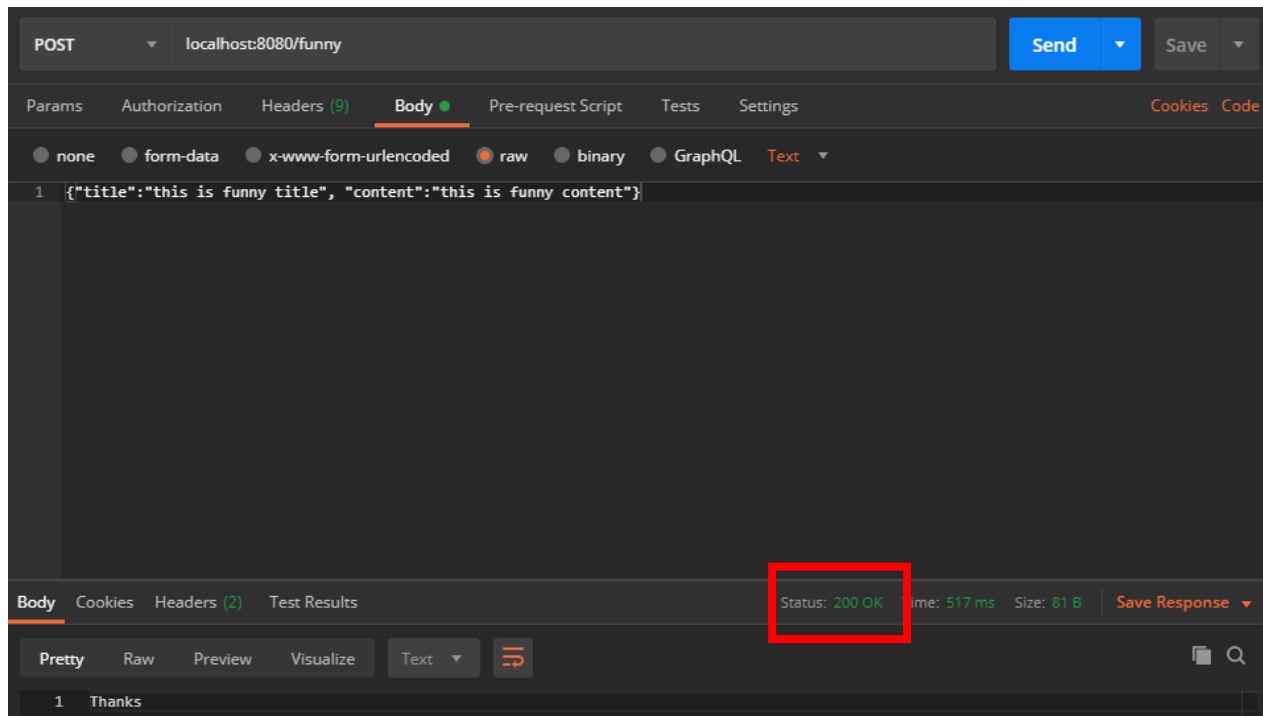
시연

- 이야기 가져오기



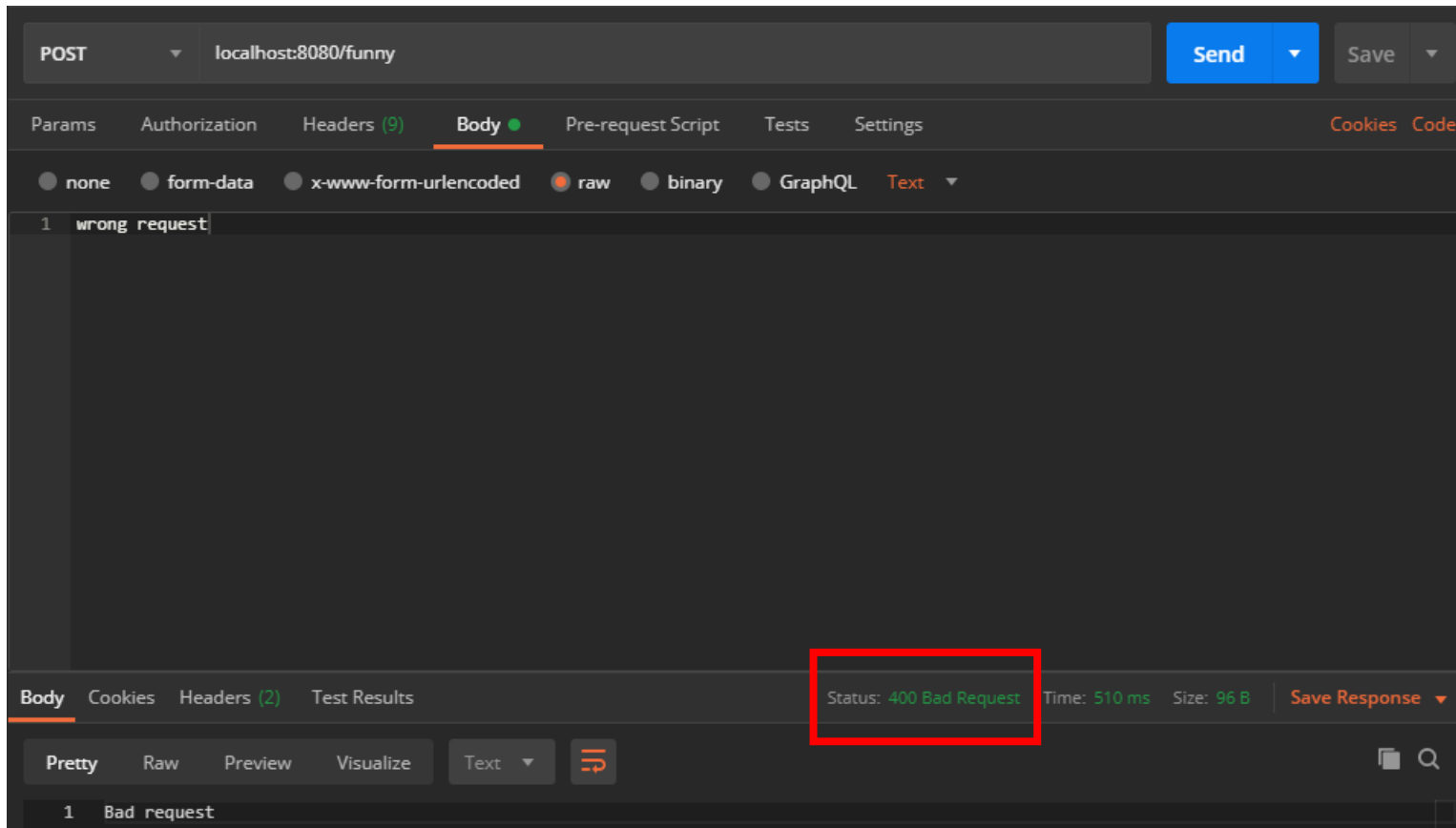
시연

- 이야기 공유하기 - 성공



시연

- 이야기 공유하기 - 실패



배운 점

- Interface, Abstract class

Interface, Abstract class 를 통한 추상화의 장점을 이해하고 구현해보았다.

- Web framework의 작동 방식

1. Web server 를 구축할 때 필요한 기능 이해하였다.
2. Handler 를 통해 다양한 방식으로 리소스를 서빙하는 방식을 배웠다.

감사합니다.