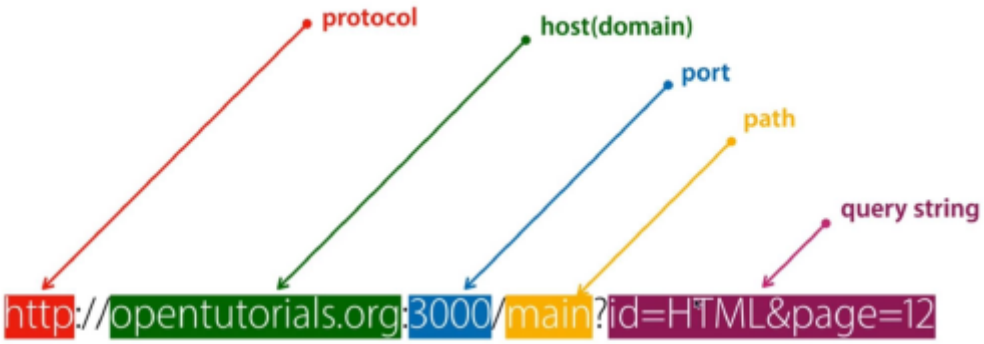
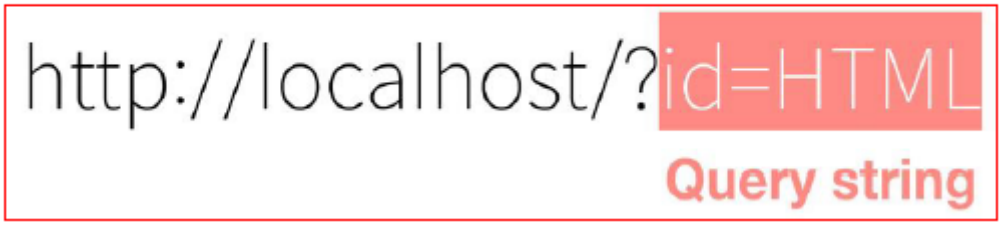
URL의 구성





- URL의 Query String에서 요청 파라미터 중 Query 값 가져오기
- url.parse(urlStr, [parseQueryString], [slashesDenoteHost])
  - url 문자열(urlStr)를 url 객체로 변환하여 리턴
  - parseQueryString과 slashesDenoteHost는 기본값으로 **false**
  - parseQueryString
    - true : url 객체의 query 속성을 **객체 형식으로 가져옴**(querystring 모듈 사용)
    - false : url 객체의 query 속성을 문자열 형식으로 가져옴
  - slashesDenoteHost
    - true : urlStr 이 '//foo/bar'인 경우 foo는 host, /bar는 path로 인식
    - false : urlStr 이 '//foo/bar'인 경우 //foo/bar 전체를 path로 인식하고 host는 null
- URL의 query값 가져오기
  - **url.parse(urlStr, true).query**
  - 쿼리스트링의 id 속성 값으로 가져오기

파일 시스템 접근하기

fs

- 파일 시스템에 접근하는 모듈
  - 파일/폴더 생성, 삭제, 읽기, 쓰기 가능
  - 웹 브라우저에서는 제한적이었으나 노드는 권한을 가지고 있음
  - 파일 읽기 예제

```
const fs = require('fs');

fs.readFile('./readme.txt', (err, data) => {
    if (err) {
        throw err;
    }

    console.log(data);
    console.log(data.toString());
});
```

fs로 파일 만들기

- 파일을 만드는 예제

```
const fs = require('fs');

fs.writeFile('./writeme.txt', '글이 입력됩니다', (err) => {
    if (err) {
        throw err;
    }
    fs.readFile('./writeme.txt', (err, data) => {
        if (err) {
            throw err;
        }
        console.log(data.toString());
    });
});
```

동기 메서드와 비동기 메서드

- 노드는 대부분의 내장 모듈 메서드를 비동기 방식으로 처리
  - 비동기는 **코드의 순서와 실행 순서가 일치하지 않는 것**을 의미
  - 일부는 동기 방식으로 사용 가능

```
const fs = require('fs');

console.log('시작');
fs.readFile('./readme2.txt', (err, data) => {
    if (err) {
        throw err;
    }
    console.log('1번', data.toString());
});
fs.readFile('./readme2.txt', (err, data) => {
    if (err) {
        throw err;
    }
    console.log('2번', data.toString());
});
fs.readFile('./readme2.txt', (err, data) => {
    if (err) {
        throw err;
    }
    console.log('3번', data.toString());
```

```
});

console.log('끝');
```

  - 매번 순서가 다르게 실행됨
- 동기 메서드 사용하기

```
const fs = require('fs');

console.log('시작');
let data = fs.readFileSync('./readme2.txt');
console.log('1번', data.toString());
data = fs.readFileSync('./readme2.txt');
console.log('2번', data.toString());
data = fs.readFileSync('./readme2.txt');
console.log('3번', data.toString());
console.log('끝');
```

- 비동기 메서드로 순서 유지하기

```
const fs = require('fs');

console.log('시작');
fs.readFile('./readme2.txt', (err, data) => {
    if (err) {
        throw err;
    }
    console.log('1번', data.toString());
    fs.readFile('./readme2.txt', (err, data) => {
        if (err) {
            throw err;
        }
        console.log('2번', data.toString());
        fs.readFile('./readme2.txt', (err, data) => {
            if (err) {
                throw err;
            }
            console.log('3번', data.toString());
        });
    });
});
console.log('끝');
```

  - 코드가 너무 우측으로 너무 들어가는 현상 발생(Callback hell)
- fs 파일 시스템에 접근하는 모듈
  - 파일/폴더 생성, 삭제, 읽기, 쓰기 가능
  - 웹 브라우저에서는 제한적이었으나 논드는 권한을 가지고 잇음
  - 형식) fs.readFile(filename, [options], callback);
  - Filename : 동적으로 변하도록 처리
  - options : utf-8 지정시 한글처리