SeSAC 용산 1기,

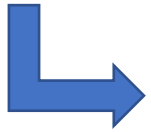
MVC 수업

WITH 팀 뤼쳐드

MVC

MVC란?

- **Model View Controller**
- 소프트웨어 설계와 관련된 **디자인 패턴**
- MVC 이용 웹 프레임워크
  - PHP
  - Django
  - **Express**
  - Angular 등등

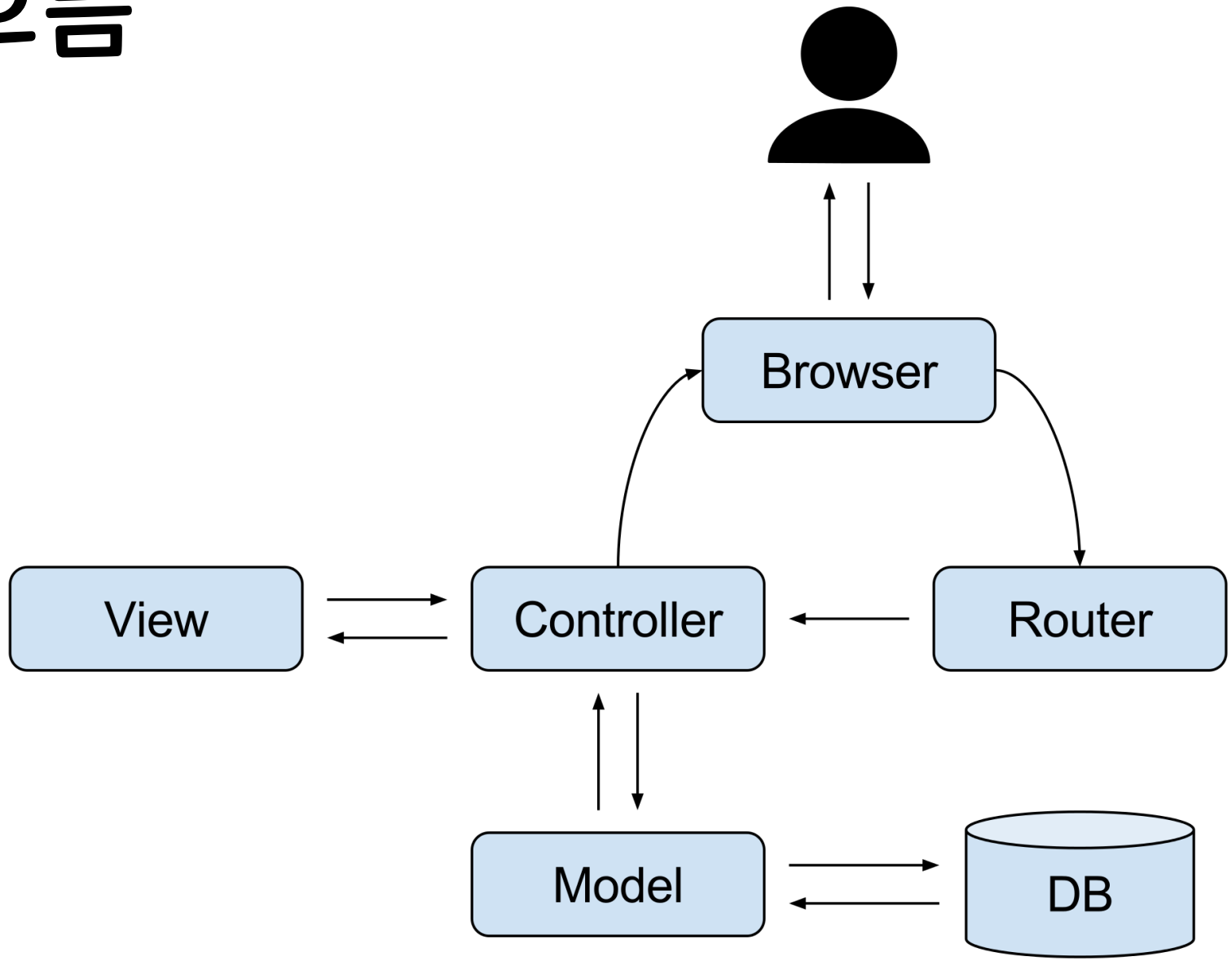


상황에 따라 자주 쓰이는 설계 방법을 정리한 코딩 방법론!!

MVC 장단점

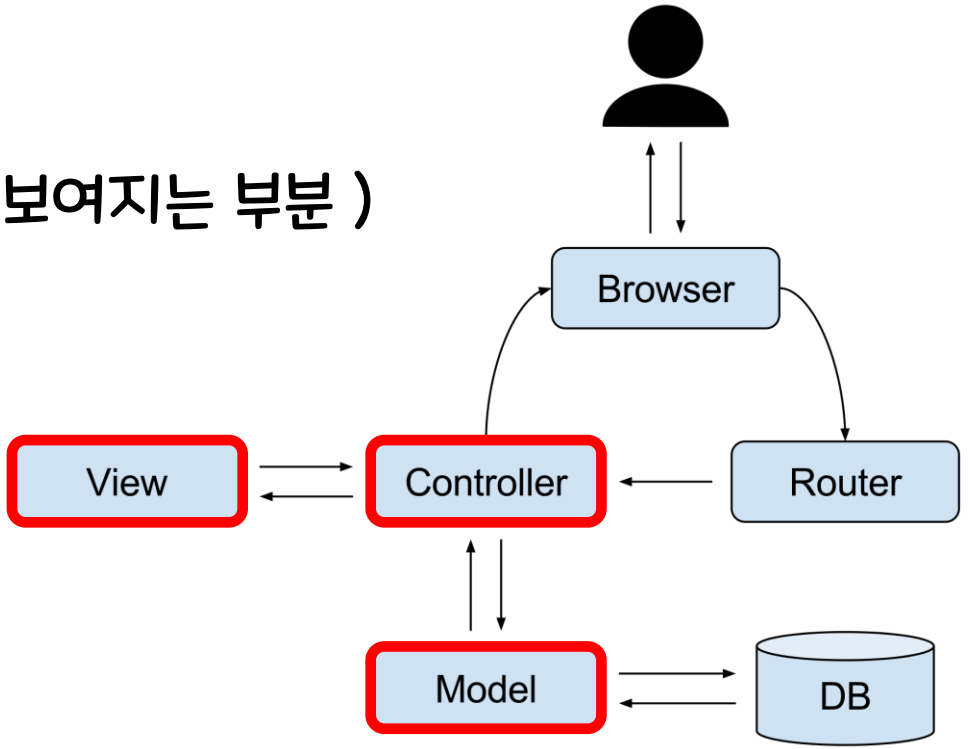
- 장점
  - 패턴들을 구분해 개발한다.
  - 유지보수가 용이하다.
  - 유연성이 높다.
  - 확장성이 높다.
  - 협업에 용이하다.
- 단점
  - 완벽한 의존성 분리가 어렵다.
  - 설계 단계가 복잡하다.
  - 설계 시간이 오래 걸린다.
  - 클래스가 많아진다.

MVC 흐름



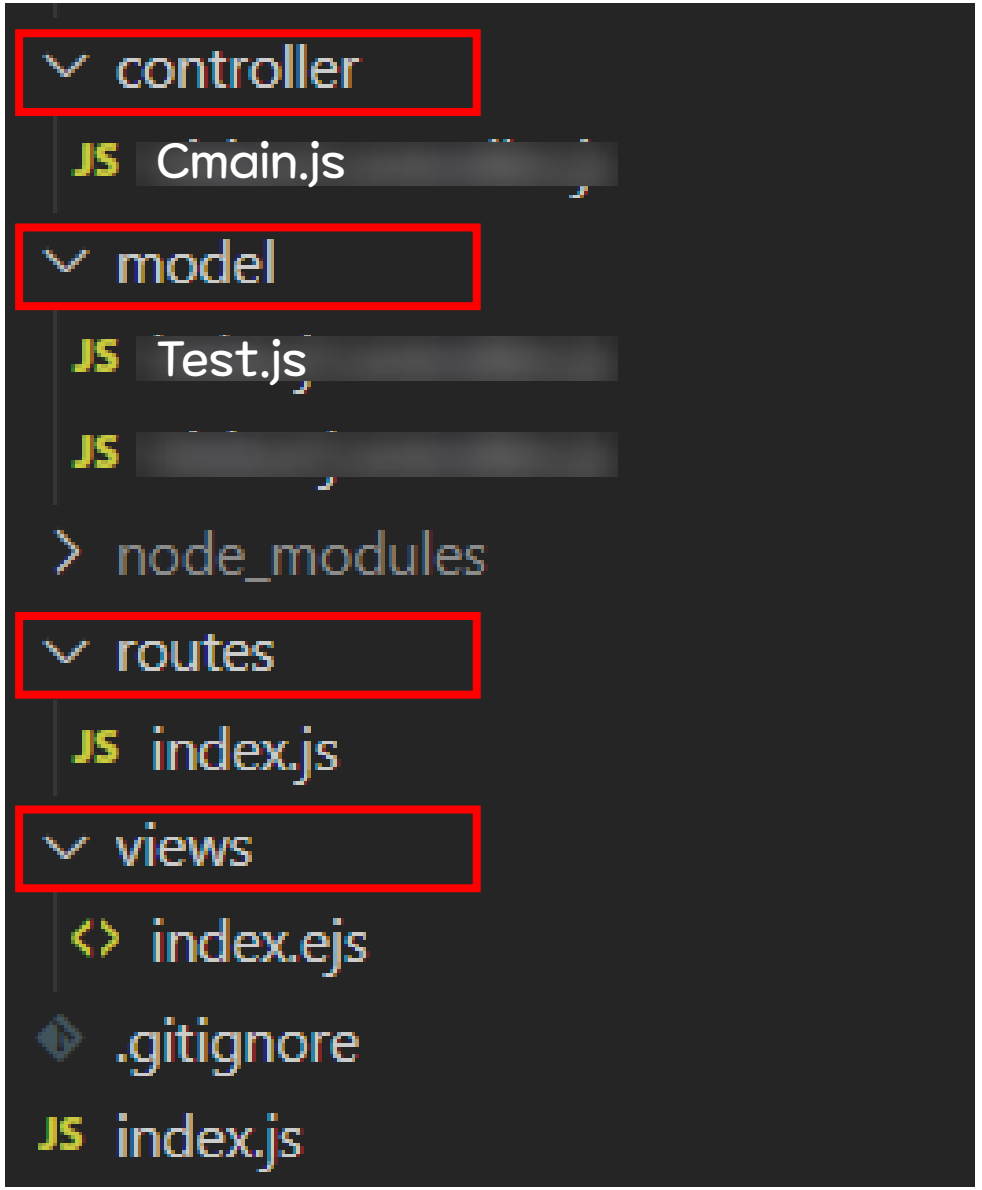
MVC 흐름

- **Model**
  - 데이터를 처리하는 부분
- **View**
  - UI 관련된 것을 처리하는 부분 (사용자에게 보여지는 부분)
- **Controller**
  - View 와 Model을 연결해주는 부분



Node.js MVC 구조

폴더구조


controller
Cmain.js
model
Test.js
node_modules
routes
index.js
views
index.ejs
.gitignore
index.js

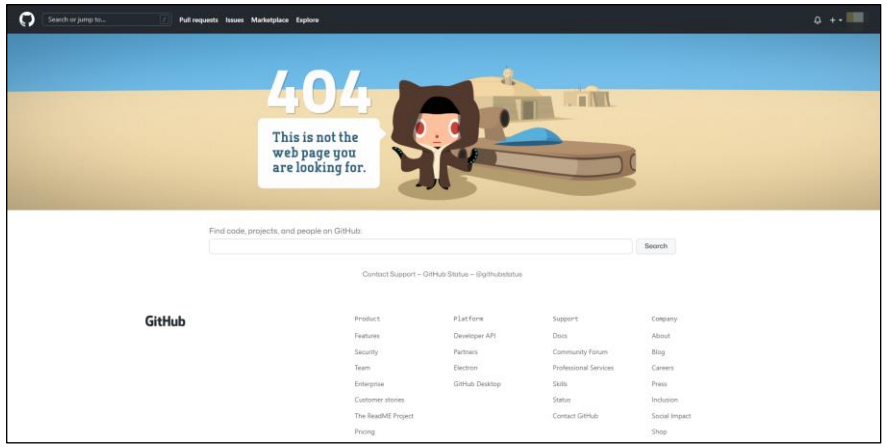
index.js

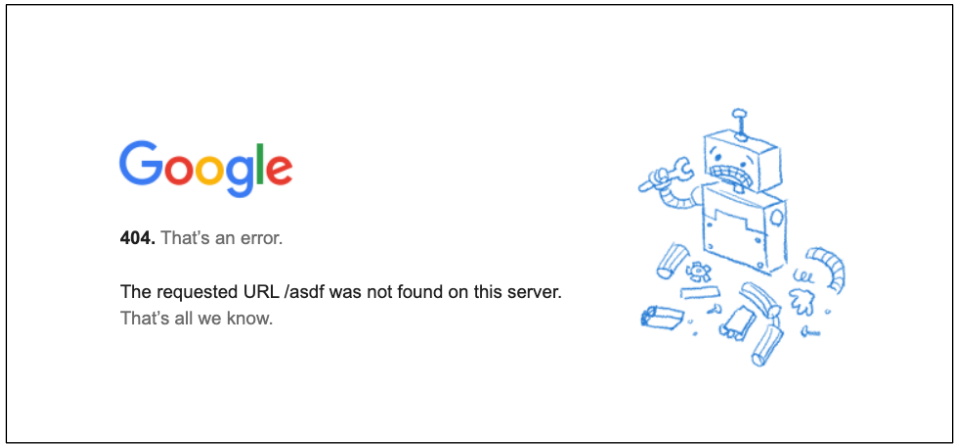
```
const express = require("express");
const app = express();
const port = 8000;

app.set("view engine", "ejs");

app.use("/static", express.static(__dirname+"/static"));
app.use(express.urlencoded({ extended: false }));
app.use(express.json());

const router = require("./routes");
app.use('/', router);

app.get('*', (req, res)=>{
    res.render('error');
})

app.listen(port, ()=>{
    console.log("server open: ", port);
});
```

- Router 을 불러오는 부분
- 위의 코드를 이용해 특정 시작 url의 역할을 나눌 수 있다.

참고) 404 Error 란?

- 404에러는 클라이언트가 **잘못된 주소로 접속**했을 때 발생하는 Error!





예시: GitHub/Google 404 에러 화면

참고) 404 Error 라우팅

```
app.get('*', (req, res)=>{
    // res.send('404 Error! 존재하지 않는 주소입니다.')
    res.render('error');
})
```

- **맨 마지막 라우트**로 선언
- * : 그 외 나머지 주소는 모두(all) 잘못된 요청임을 사용자에게 알려야 함
- 클라이언트가 올바르지 않은 주소로 요청 시 Error 페이지 렌더링

routes/index.js

```
var express = require("express");
var controller = require("../controller/Cmain");
const router = express.Router();

router.get("/", controller.main);

module.exports = router;
```

- 경로를 controller와 연결지어 설정할 수 있다.

Controller/Cmain.js

```
exports.main = (req, res) => {
    res.send("hello");
}
```

- 경로와 연결될 함수 내용을 정의한다.
- 경로와 연결되는 함수이기에 request 객체와 response 객체를 사용할 수 있다.

model/Test.js

```
exports.hello = function(){
    return "hello";
}
```

- (임시) DB에서 “hello”라는 데이터를 받았다고 가정하고 코드를 작성한 것

Controller - model 연결

```
const Test = require("../model/Test");

exports.main = (req, res) => {
    var hi = Test.hello();
    res.send(hi);
}
```

- 컨트롤러와 모델을 연결한다.