■ Router 객체로 라우팅 분리하기

: express → 깔끔한 라우팅 관리기능 제공 및 라우터를 분리할 수 있게 돕는다.

: app.get 등의 메서드가 라우터 부분이다.

예.) routes/index.js

```
const express = require('express');
const router = express.Router();  //interface

// GET/라우터
router.get('/',(req, res)=>{
    res.send('Hello, Express');
});

module.exports = router;
```

routes/user.js

```
const express = require('express');
const router = express.Router();

// GET/user 라우터
router.get('/', (req, res)=>{
    res.send('Hello, User');
});

module.exports = router;
```

라우터 연결

app.js

```
//==>  router 분리를 하기 위한 설정.===

const indexRouter = require('./routes'); //'./routes/index.js' 로 라우팅
const userRouter = require('./routes/user'); //'./routes/user.js' 로 라우팅
//=====================================
const app = express();                    //== 기본
app.set('port', process.env.PORT||3000);//== 기본 http://localhost:3000
//======> 라우터 연결
app.use('/', indexRouter);      //indexRouter 는 app.use('/')에 연결
app.use('/user', userRouter);  //userRouter 는 app.use('/user')에 연결

app.use((req, res, next)=>{
    res.status(404).send('Not Found');
});
//==============================
```