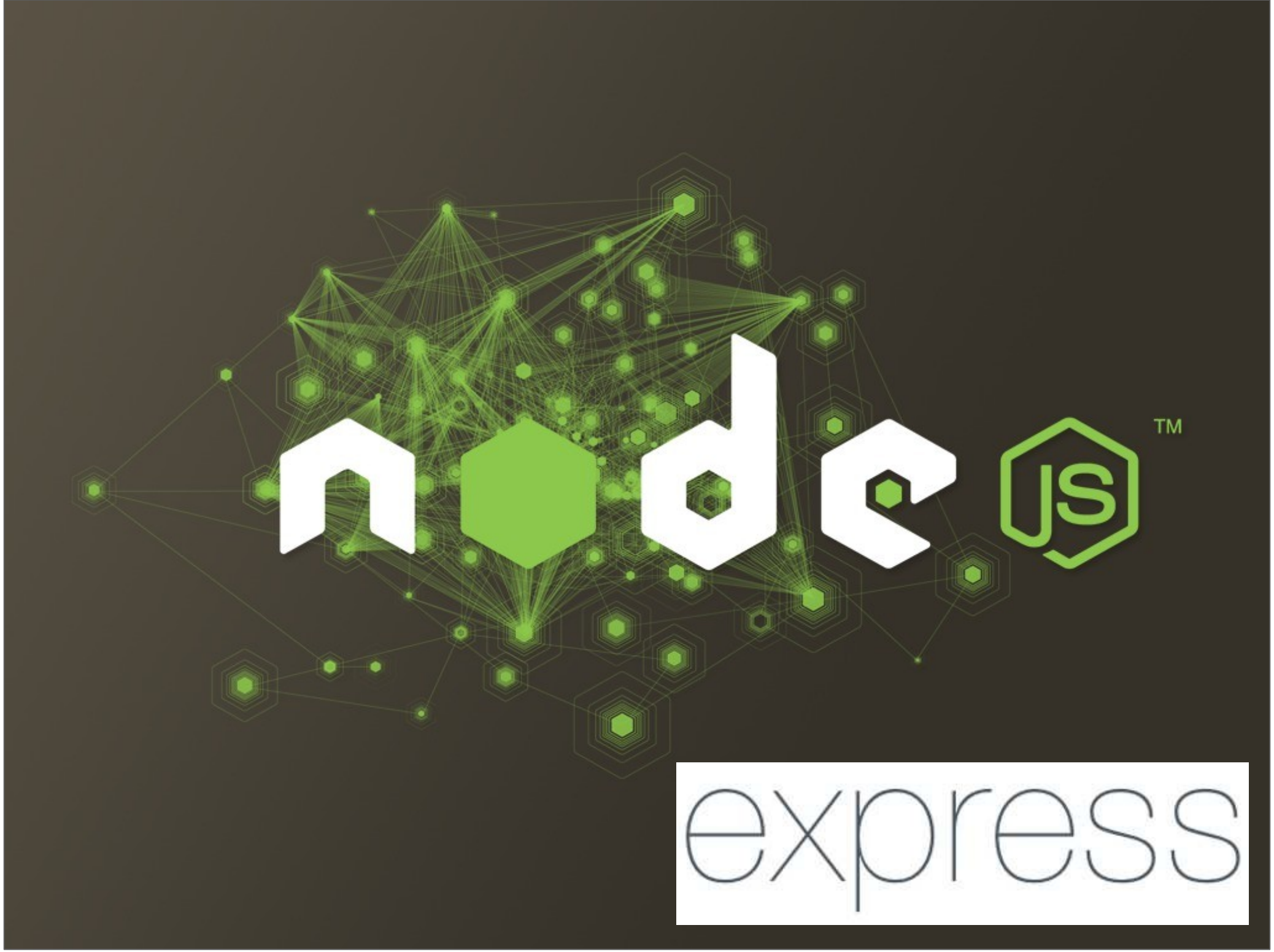
PEMROGRAMAN WEB LANJUT

EXPRESS

MINGGU KE–12

Institut Teknologi Sumatera


node
JS
TM
express

Agenda

- Express
  - About Express
  - Installing Express & Generate an app
  - Middleware: Express
  - Express Features
  - Appendix

ABOUT EXPRESS

About

- is a minimal and flexible web application framework
- provide a robust set of features for building single, multi-page and hybrid web applications
- is based on the Connect middleware
- and implements MVC

What is Express?

It is a node.js based web framework

It's the equivalent to a Servlet + web.xml in Java!

```
npm install --save express
```

Philosophy behind ExpressJS

This is probably one of the most popular frameworks in the Node.js community of coders right now. The core philosophy behind *ExpressJS* is to offer an application framework for single or multi-page web apps, using views and *template engine*. The Express.js framework also includes a multitude of utilities to work with HTTP and building APIs. The pluggable middleware in the framework provides a great way to add elements, such as *passportjs*, as you need them.

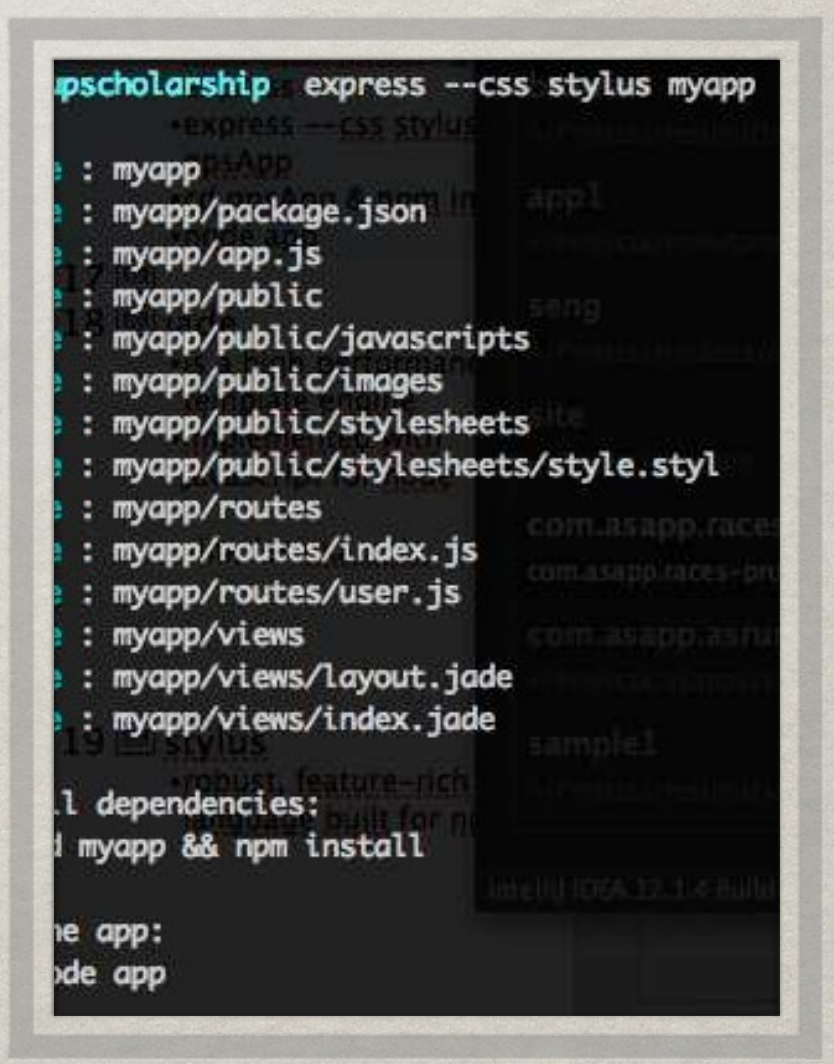
Installing & Generate an app

Installing & Generate an app

- sudo npm install -g express
- express --sessions --css stylus myapp
- cd myapp & npm install
- node app

Express - app

- package.json (some app settings)
- app.js (launch app)
- public/*
- routes/*
- views/*



Middleware: Express

Middleware: Express

Express does to Connect what Connect does to the http module: It offers a *createServer* method that extends Connect's Server prototype. So all of the functionality of Connect is there,plus view rendering and a handy DSL for describing routes.

Express - Connect / Middleware

- Connect is an extensible HTTP server framework for node, providing high performance "plugins" known as middleware.

- A middleware is simply a function with three arguments: request, response, next

```
function uselessMiddleware(req, res, next) { next() }

// A middleware that simply interrupts every request
function worseThanUselessMiddleware(req, res, next) {
  next("Hey are you busy?")
}
```


error -> request interruption

Express - Connect / Middleware

```
// a middleware mounted on /user/:id; will be executed for any type of HTTP
request to /user/:id
app.use('/user/:id', function (req, res, next)
  {  console.log('Request Type:', req.method);
  next();
})


// a middleware with no mount path; gets executed for every request to the app
app.use(function (req, res, next) {
  console.log('Time:', Date.now());
  next();
});
```

Express Features

Express - HTTP Server

```
var express = require('express');
var http = require('http');



//create express app

var app = express();

app.set('port', process.env.PORT || 3000);



http.createServer(app).listen(app.get('port'), function()
  {  console.log('Express server listening on port ' +
  app.get('port'));
});
```

Express - Routing

Routing is a way to map different requests to specific handlers.

```
app.get('/', function(request, response)
    {  response.send('¡Hello, Express!');
});

app.get('/users/:userName', function(request, response)
    {  var name = request.params.userName;
    response.send('¡Hello, ' + name + '!');
});

//regex

app.get(/\/users\/(\d*)\/?(edit)?/, function (request, response)
    {  var message = 'user number #' + request.params[0];
    if (request.params[1] === 'edit')
        {  message = Editing ' +
        message;
    } else {
        message = 'Viewing ' + message;
    }
    response.send(message);
});
```

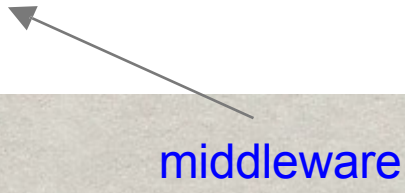
handler

Express - Routing POST

```
app.post('/users', function(request, response)
    {  var username = request.body.username;
    response.send('¡Hello, ' + username + '!');
});
```

Express doesn't parse request body by default

```
app.use(express.bodyParser());
```



Express - Request Handling

Express augments the request and response objects that you're passed in every request handler.

```
response.redirect("/hello/anime");
response.redirect("http://www.myanimelist.net");
response.redirect(301, "http://www.anime.org");     // HTTP status code 301
```

This isn't in vanilla Node and it's also absent from Connect, but Express adds this stuff. It adds things like sendFile which lets you just send a whole file:

```
response.sendFile("/path/to/anime.mp4");
```

The request gets a number of cool properties, like request.ip to get the IP address and request.files to get uploaded files.

Express - View rendering

More features? Oh, Express, I'm blushing.
Express can handle views. It's not too bad. Here's what the setup looks like:

```
app.set('views', path.join(_dirname, '..', '..', 'templates'));

app.set('view engine', 'dust');
```

```
// Start Express
var express = require("express");  var app = express();

// Set the view directory to /views  app.set("views",_dirname
+ "/views");

// Let's use the Jade templating language  app.set("view
engine", "jade");
```

The first block is the same as always. Then we say "our views are in a folder called 'views'".

Everything from Connect and Node

I want to remind you that Express is built on top of Connect which is built on top of Node. This means that all Connect middleware works with Express. This is useful! For example:

```
var express = require("express");  var app = express();

app.use(express.logger());  // Inherited from Connect  app.get("/", function(req, res) {
  res.send("anime");
});

app.listen(1337);
```

Express - Error Handling

- Error-handling middleware always takes ***four*** arguments.

```
app.use(function(err, req, res, next)
  {  console.error(err.stack);
  res.status(500).send('Something broke!');
});
```

Express - Official middlewares

These middleware and libraries are officially supported by the Connect/Express team:

- body-parser - previous `bodyParser`, `json`, and `urlencoded`. You may also be interested in:
  - body
  - co-body
  - raw-body
- compression - previously `compress`
- connect-timeout - previously `timeout`
- cookie-parser - previously `cookieParser`
- cookie-session - previously `cookieSession`
- csurf - previously `csrf`
- errorhandler - previously `error-handler`
- express-session - previously `session`
- method-override - previously `method-override`
- morgan - previously `logger`
- response-time - previously `response-time`
- serve-favicon - previously `favicon`
- serve-index - previously `directory`
- serve-static - previously `static`
- vhost - previously `vhost`

Appendix

Appendix

- Express
  - http://expressjs.com/ http://www.senchalabs.org/connect/ http://learnboost.github.io/stylus
  - http://jade-lang.com/
  - http://stackoverflow.com/questions/5284340/what-is-node-js connect express- and middleware
  - http://expressjs.com/guide.html https://github.com/senchalabs/connect
  - http://expressjs.com/2x/guide.html#middleware


obrigado
Dank U
Merci
mahalo
Köszi
спасибо
Grazie
Thank you
mauruuru
Takk
Gracias
Dziękuję
Děkuju
danke
Kiitos