Unit – 8 NodeJS Template engine

Introduction of template engine

- Templating engines help make our web applications dynamic in terms of data.
- At the heart of a templating engine, there is a HTML template with placeholders for dynamic data.
- At runtime, the template engine replaces variables in a template file with actual values, and transforms the template into an HTML file sent to the client.
- Template engine help us to create dynamic page content.
- This approach makes it easier to design an HTML page.
- By default, express provides us a function to send HTML files.
- Using the function we can send only static pages. If we need to inject any data in an HTML file, we need a template engine.
- Templating engines divide our code into multiple components like header, footer, body, and so on. So we can reuse any component in any layout.

popular template engines that work with Express.js/node js

1. **Pug (formerly known as jade)**
2. mustache
3. dust
4. atpl
5. eco
6. ect
7. ejs
8. haml
9. haml-coffee
10. handlebars
11. hogan
12. jazz
13. jqtpl
14. JUST
15. liquor
16. QEJS
17. swig
18. templayed
19. toffee
20. underscore
21. walrus
22. whiskers

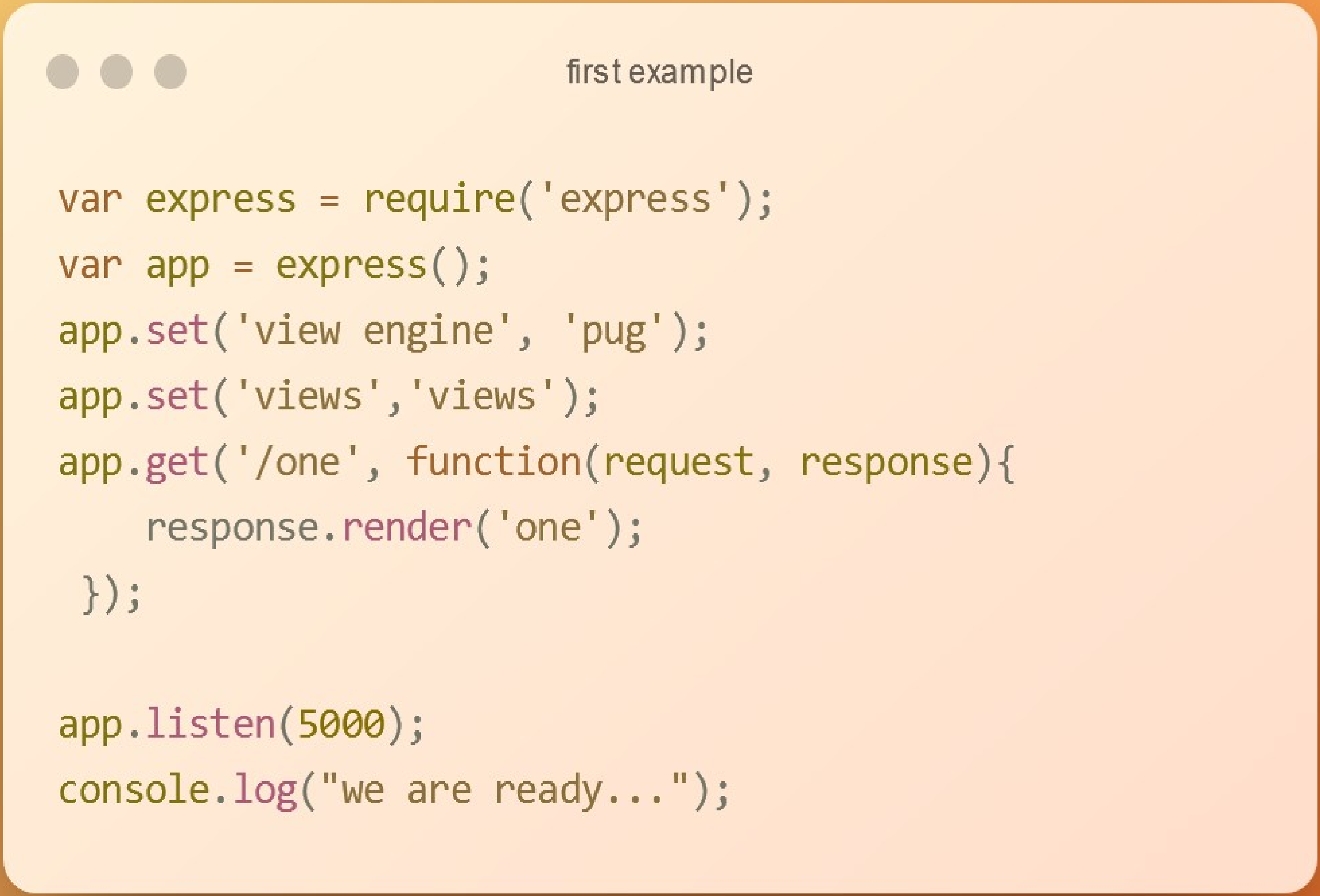
Advantages of Template engine in Node.js

1. Improves developer's productivity.
2. Improves readability and maintainability.
3. Faster performance.
4. Maximizes server side processing.
5. Single template for multiple pages.
6. Templates can be accessed from CDN (Content Delivery Network).

Introduction to pug

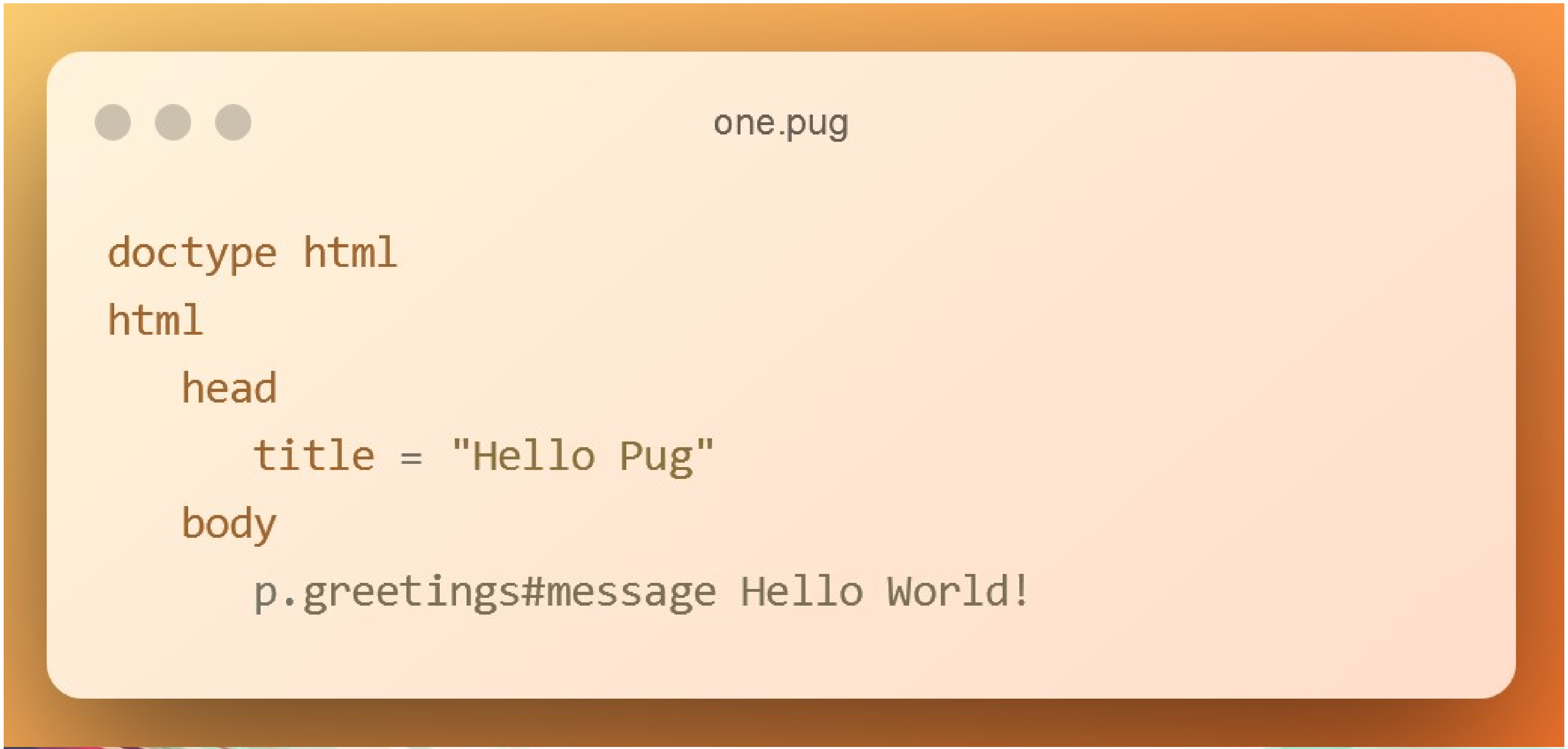
- Pug is a templating engine for Express.
- Pug is a very powerful templating engine which has a variety of features including **filters, includes, inheritance, interpolation**, etc.
- It is easy to learn and use.
- To use pug first you have to install it.
- Use below command to install pug

```
npm install --save pug
```



```
var express = require('express');
var app = express();
app.set('view engine', 'pug');
app.set('views','views');
app.get('/one', function(request, response){
    response.render('one');
 });

app.listen(5000);
console.log("we are ready...");
```



one.pug

```pug
doctype html
html
   head
      title = "Hello Pug"
   body
      p.greetings#message Hello World!
```

How it works

- Tags are nested according to their indentation. Like in the above example, **<title>** was indented within the **<head>** tag, so it was inside it. But the **<body>** tag was on the same indentation, so it was a sibling of the **<head>** tag.
- We don’t need to close tags, as soon as Pug encounters the next tag on same or outer indentation level, it closes the tag for us.

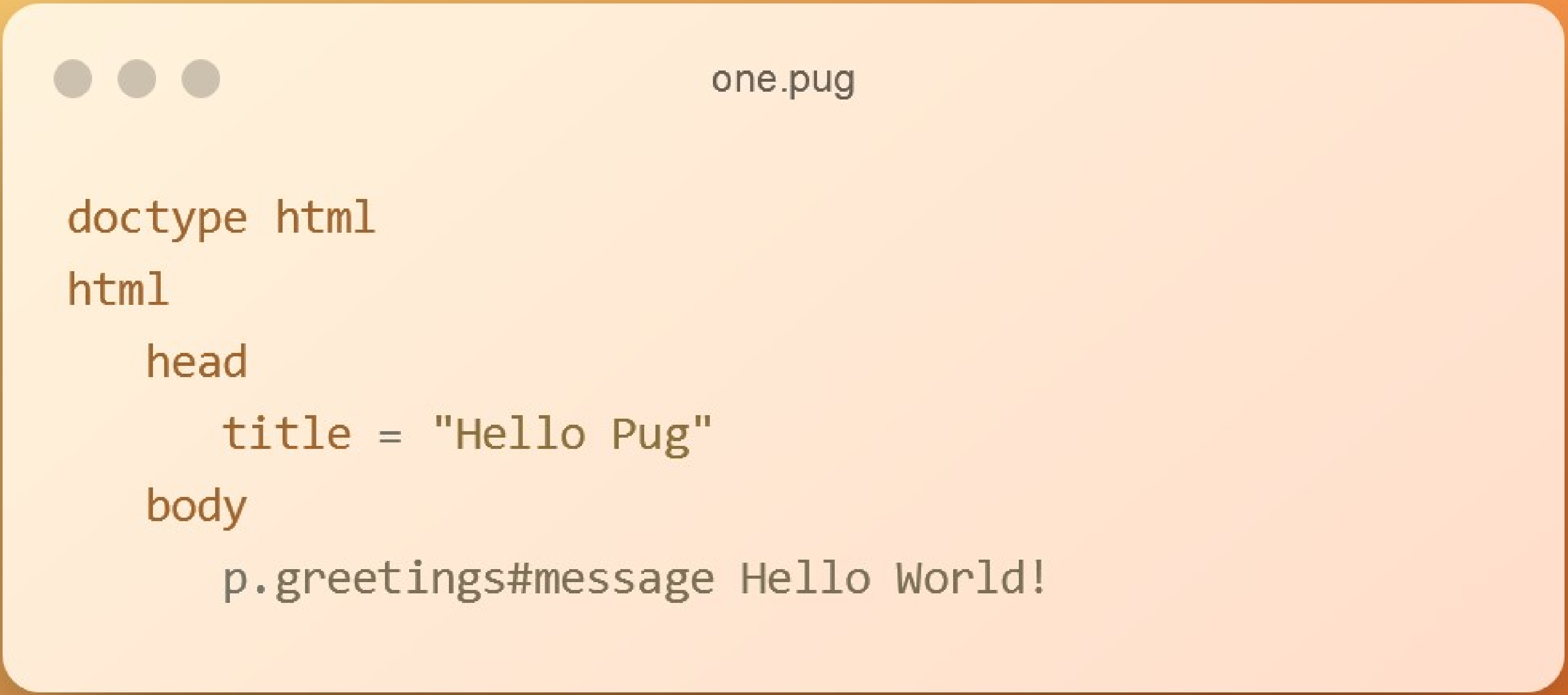
3 ways to put text inside tag

1. **Space seperated**
2. **Piped text**
3. **Block of text**

You can use online tool to convert html into pug

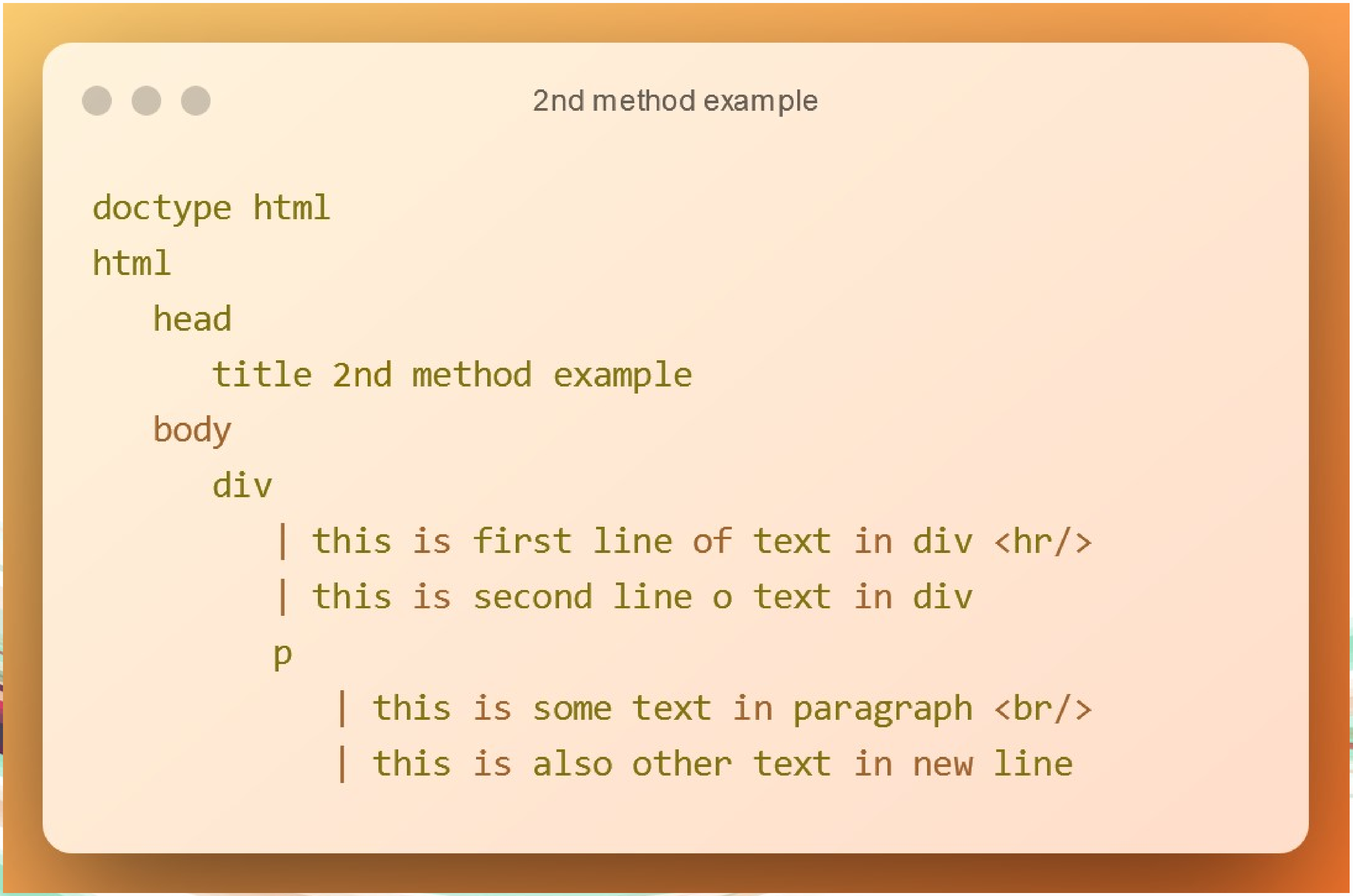
https://codebeautify.org/html-to-pug-converter

Space separated


one.pug

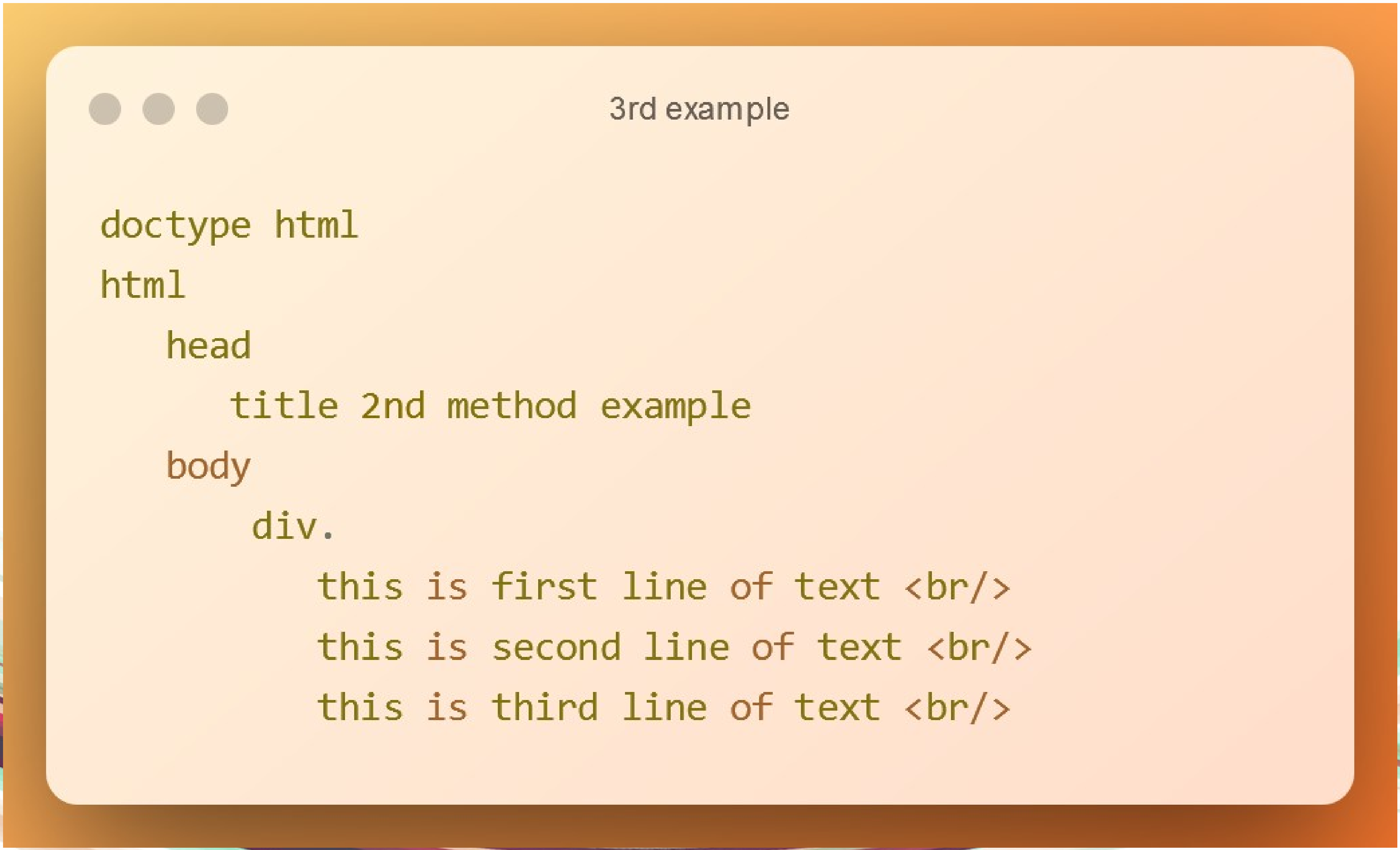
```
doctype html
html
    head
        title = "Hello Pug"
    body
        p.greetings#message Hello World!
```

Piped text



```
doctype html
html
   head
      title 2nd method example
   body
      div
         | this is first line of text in div <hr/>
         | this is second line o text in div
         p
            | this is some text in paragraph <br/>
            | this is also other text in new line
```

Block of text



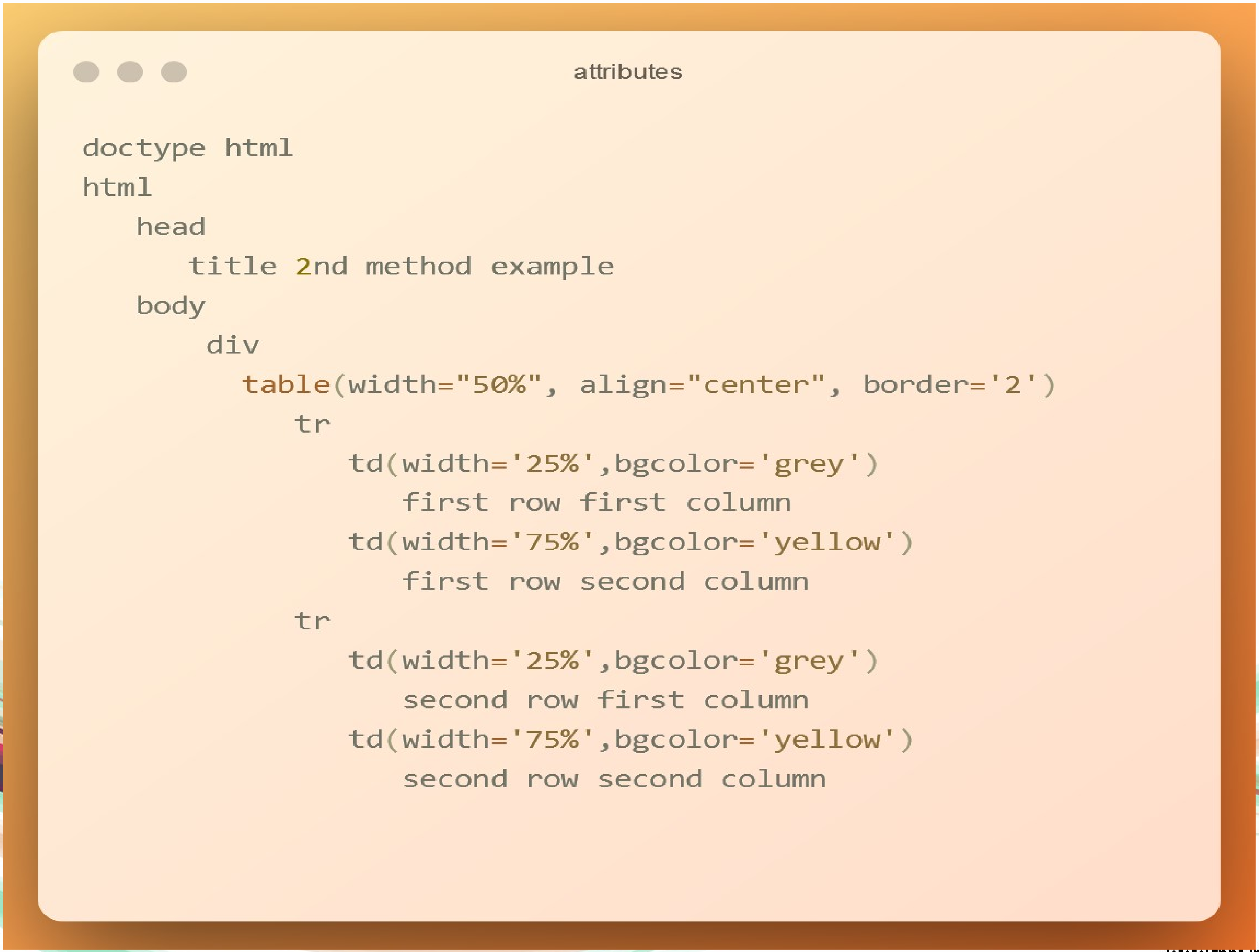
```
doctype html
html
   head
      title 2nd method example
   body
       div.
          this is first line of text <br/>
          this is second line of text <br/>
          this is third line of text <br/>
```

Comments

- We can use both type of comments in pug
- We can use single line comment using /
- We can use multiline comments using /* */
- Comments are converted into html comments when template execute.

Attributes

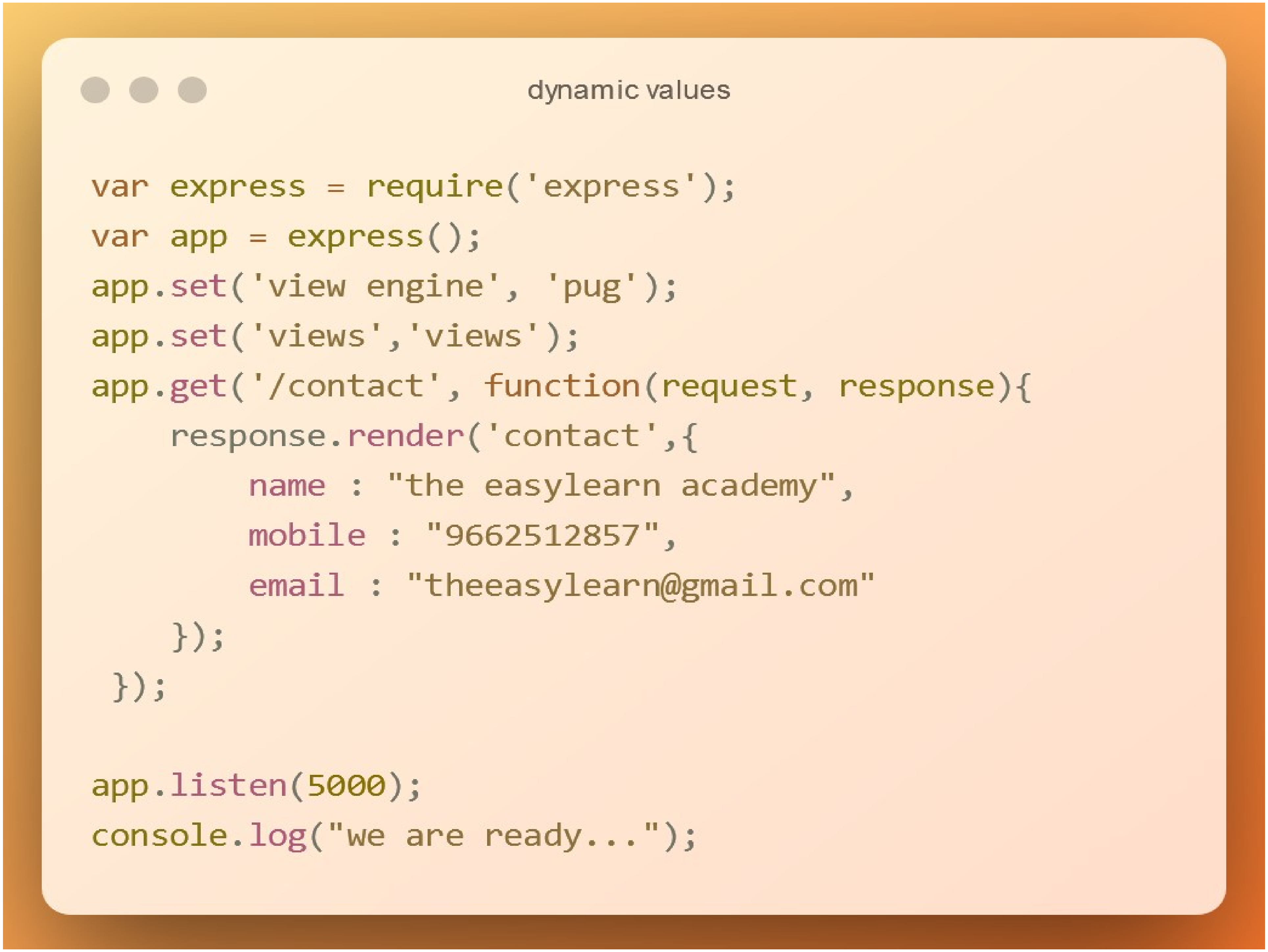
- To define attributes, we use a comma separated list of attributes, in parenthesis.
- Let us see an example



```
doctype html
html
   head
      title 2nd method example
   body
       div
         table(width="50%", align="center", border='2')
            tr
               td(width='25%',bgcolor='grey')
                  first row first column
               td(width='75%',bgcolor='yellow')
                  first row second column
            tr
               td(width='25%',bgcolor='grey')
                  second row first column
               td(width='75%',bgcolor='yellow')
                  second row second column
```

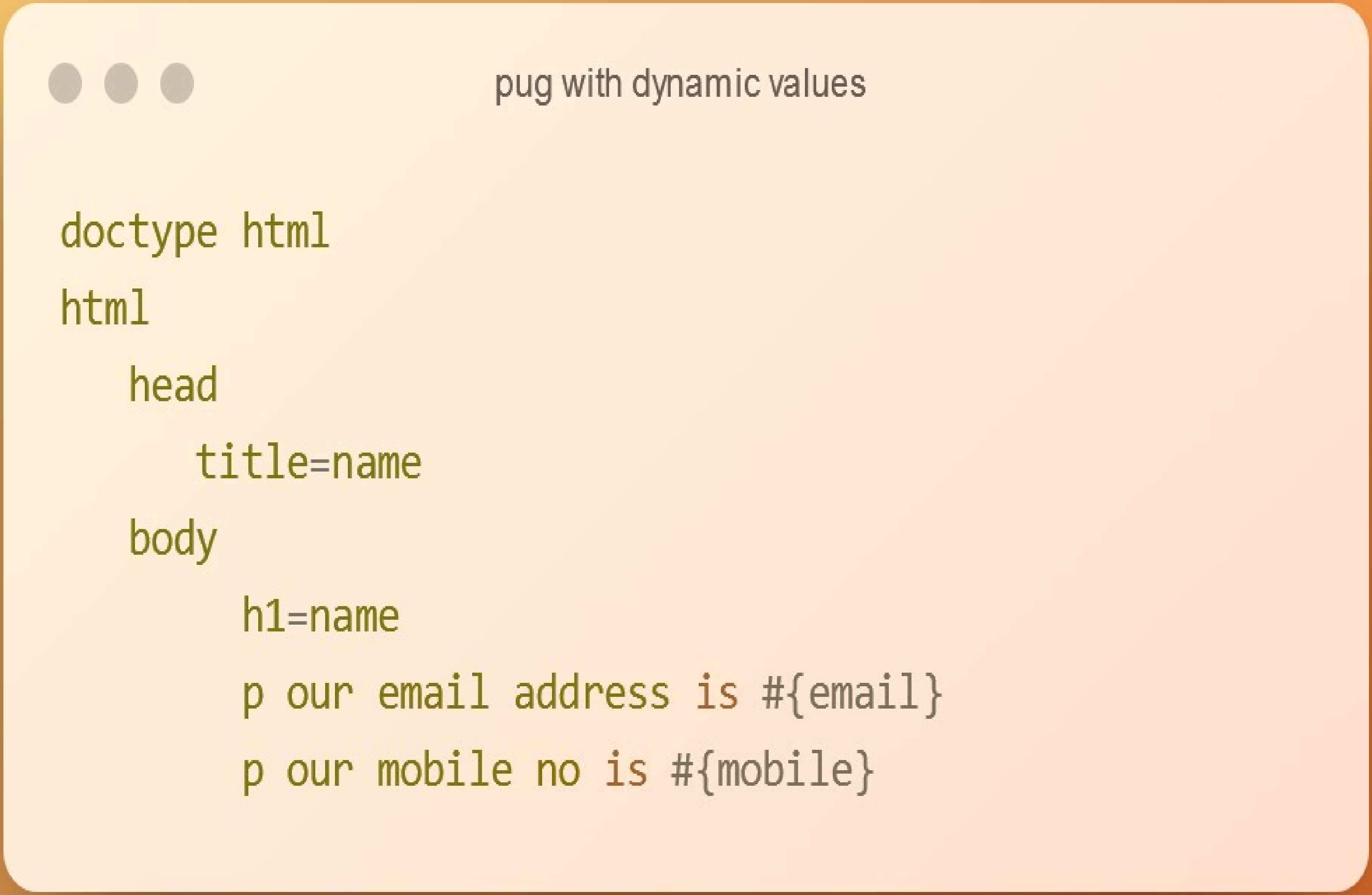
How to pass value into template?

- It is possible to pass value from route to template.
- In this way we can make our template truly dynamic.
- Let us see an example.



```
var express = require('express');
var app = express();
app.set('view engine', 'pug');
app.set('views','views');
app.get('/contact', function(request, response){
    response.render('contact',{
        name : "the easylearn academy",
        mobile : "9662512857",
        email : "theeasylearn@gmail.com"
    });
 });

app.listen(5000);
console.log("we are ready...");
```

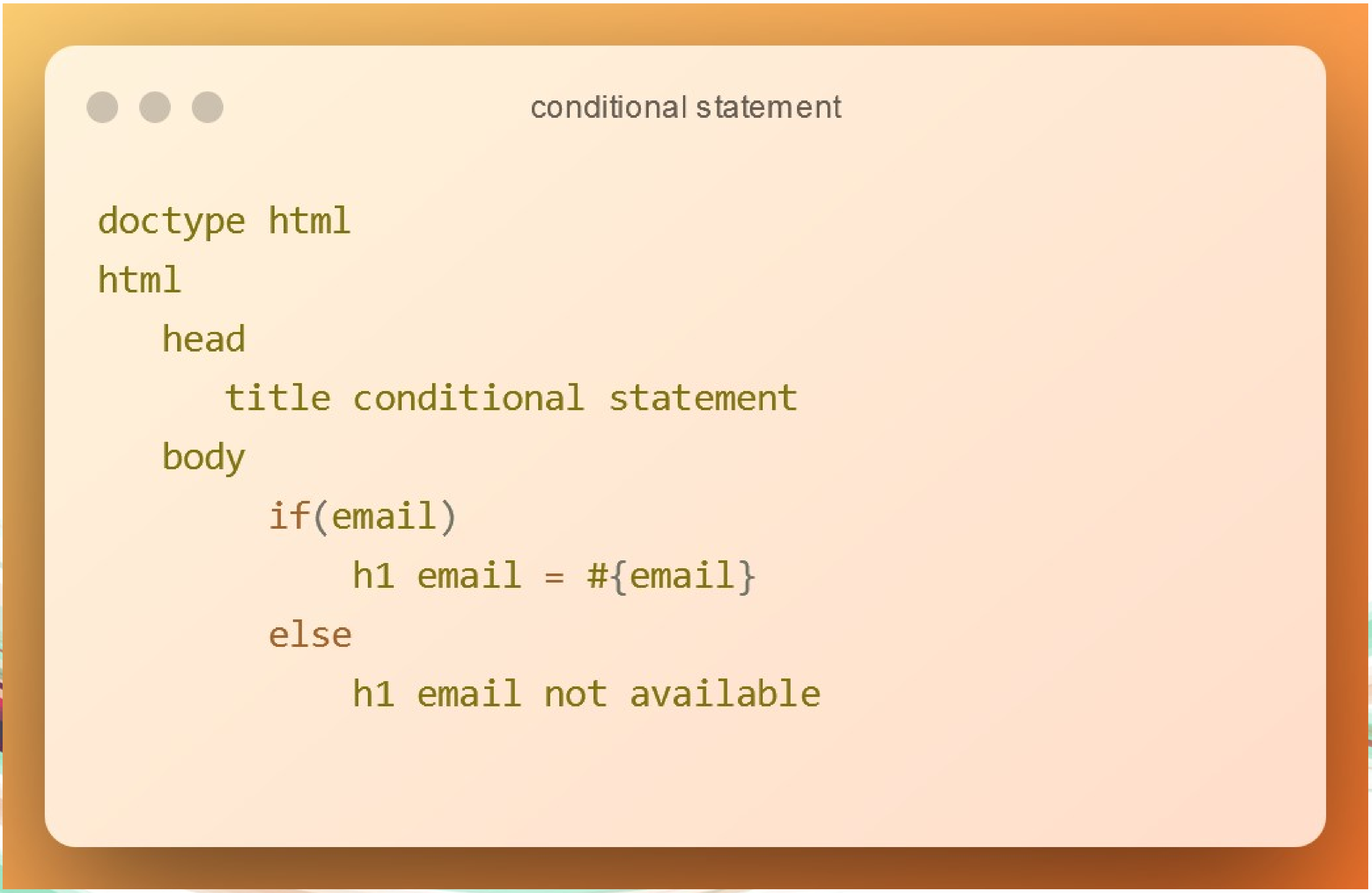


pug with dynamic values

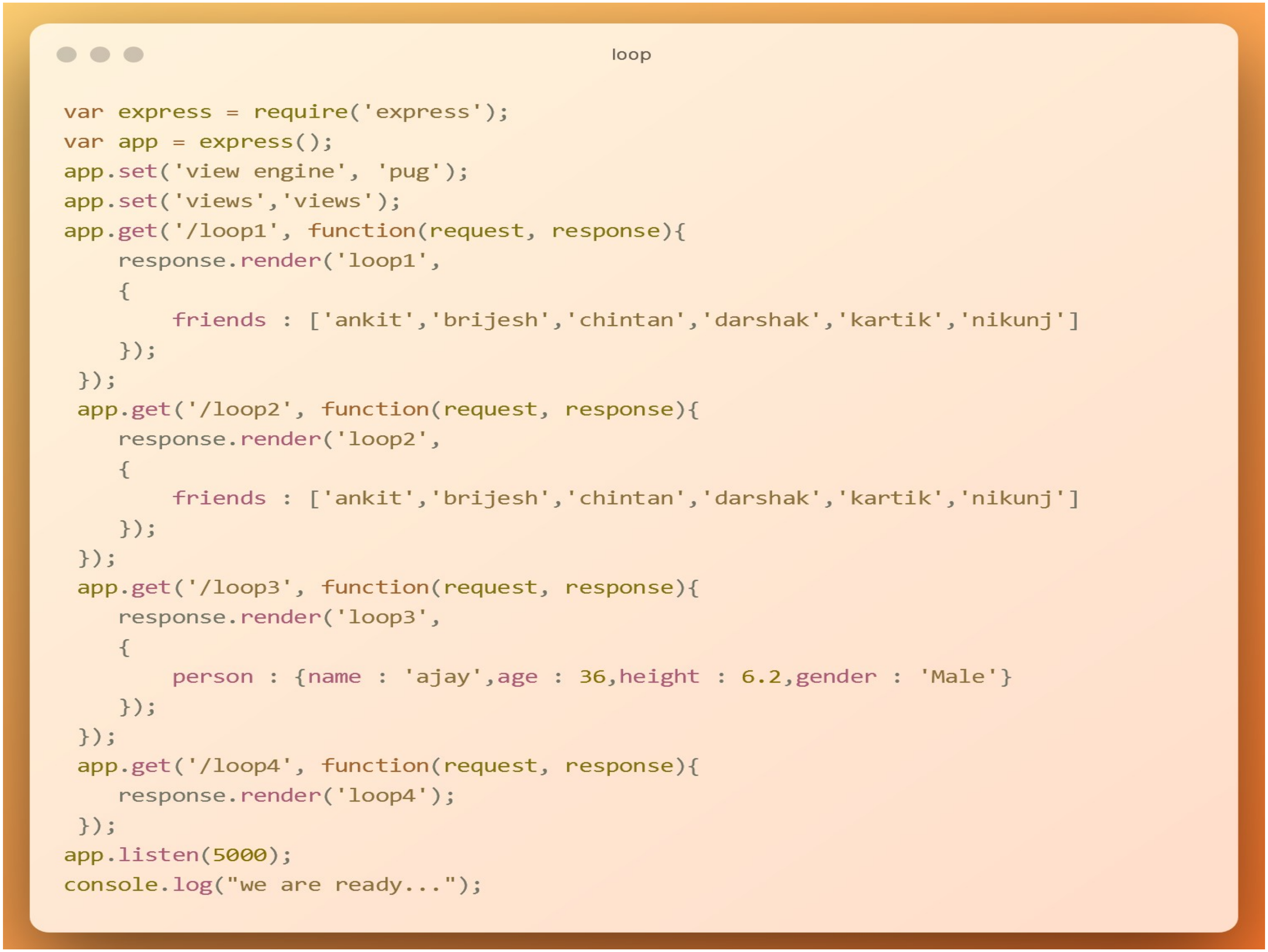
```
doctype html
html
   head
      title=name
   body
        h1=name
        p our email address is #{email}
        p our mobile no is #{mobile}
```

Conditionals

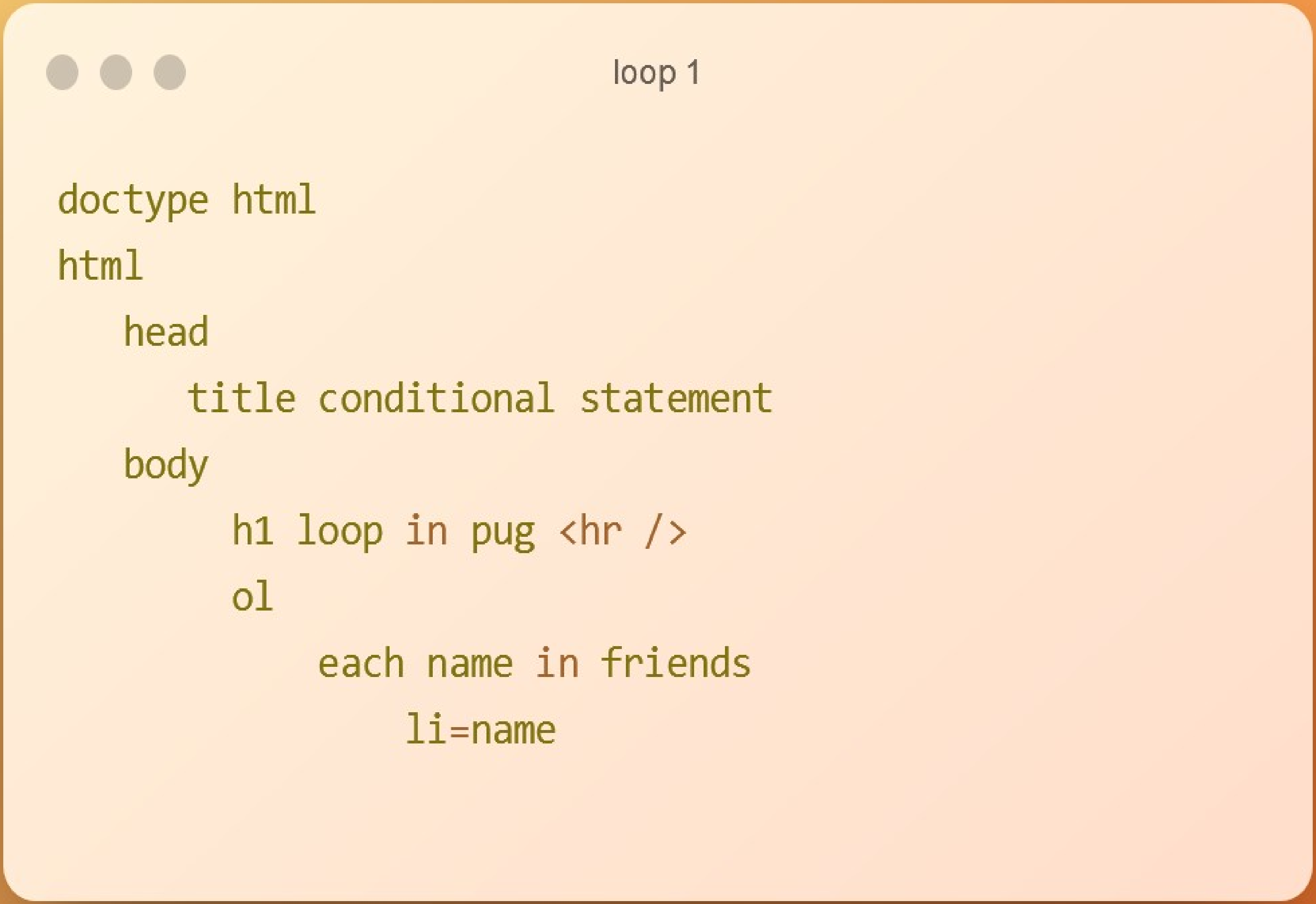
- We can use conditional statements and looping constructs as well.
- Pug supports two primary methods of iteration: each and while.



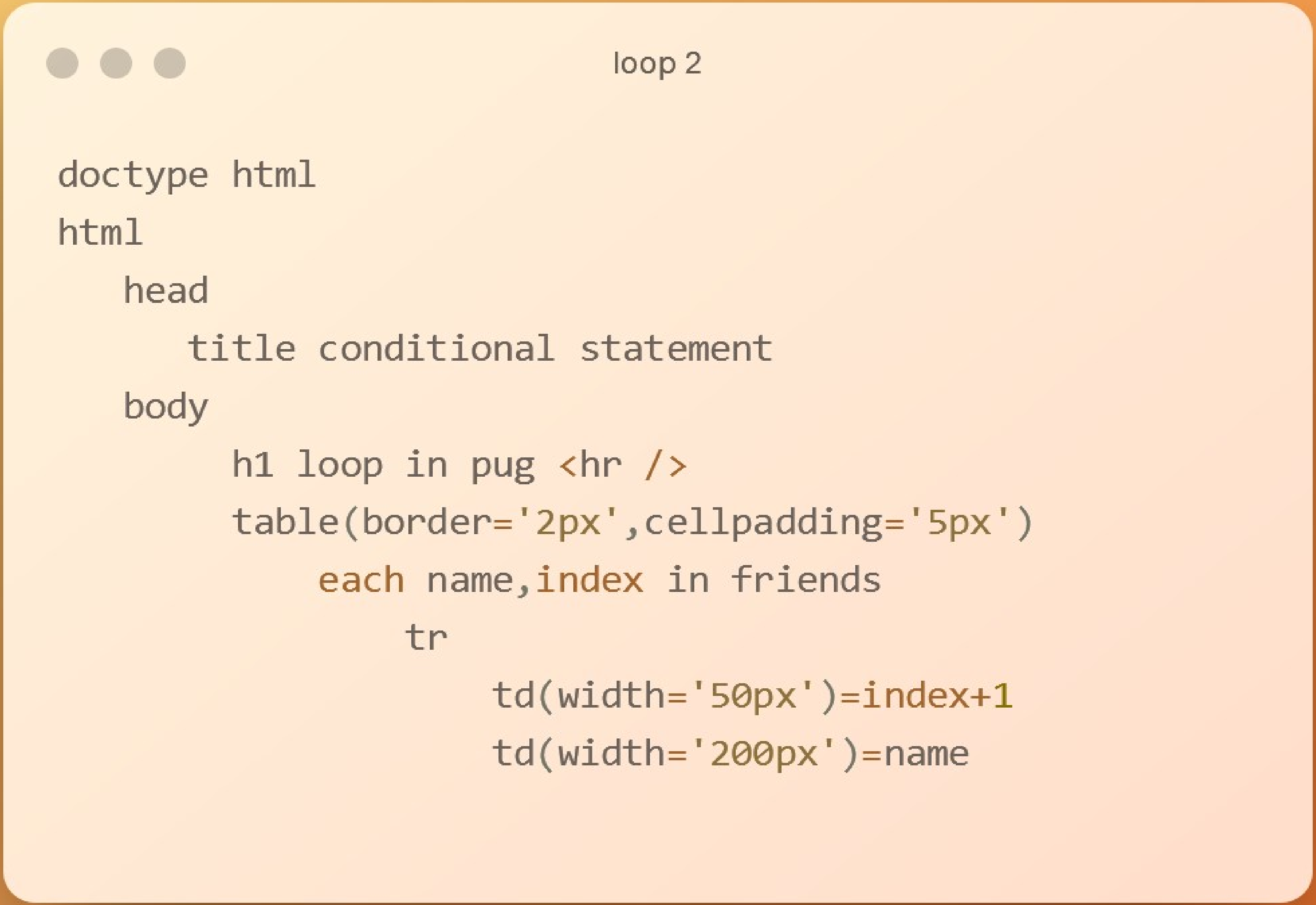
```
doctype html
html
   head
      title conditional statement
   body
        if(email)
            h1 email = #{email}
        else
            h1 email not available
```


loop

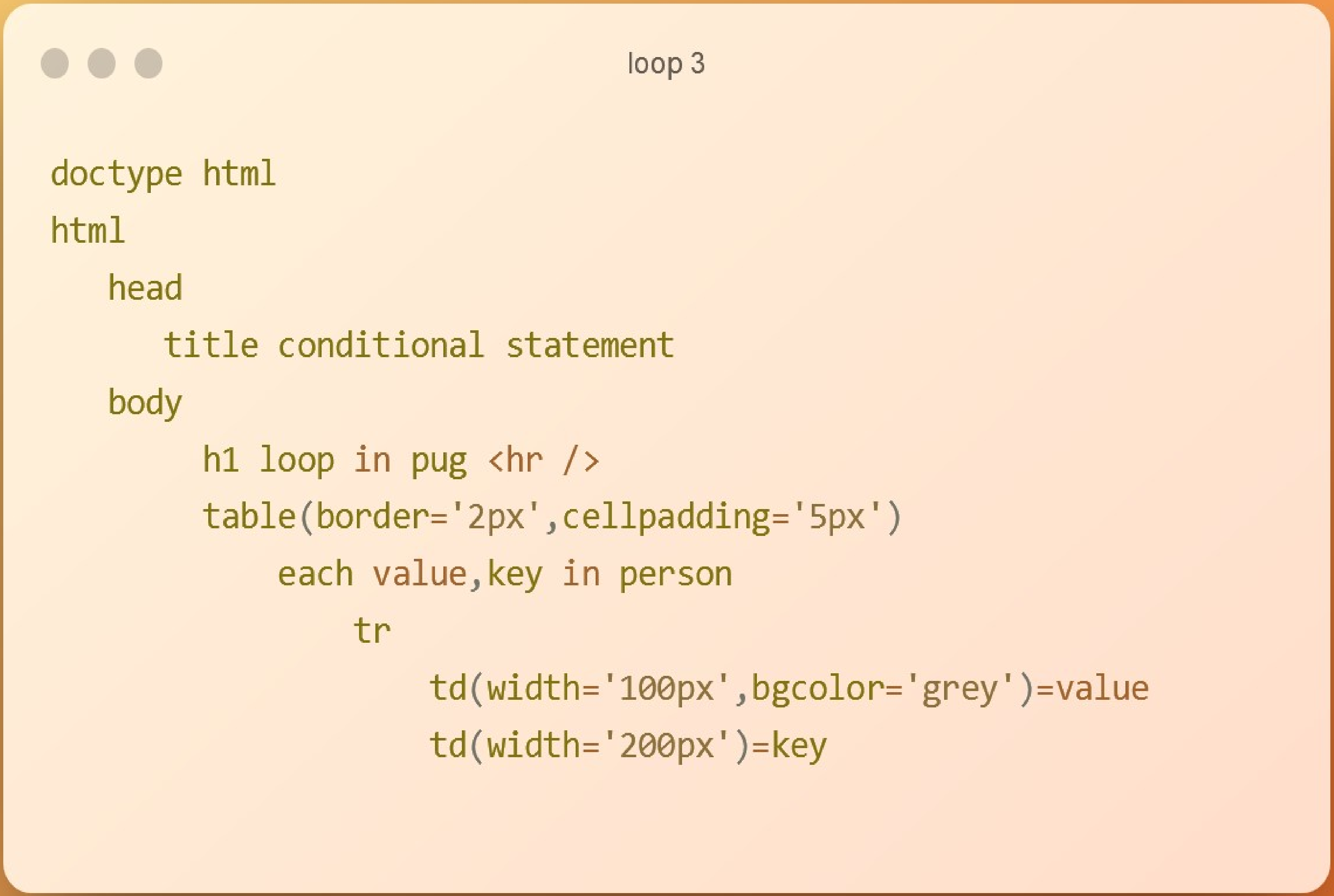
```
var express = require('express');
var app = express();
app.set('view engine', 'pug');
app.set('views','views');
app.get('/loop1', function(request, response){
    response.render('loop1',
    {
        friends : ['ankit','brijesh','chintan','darshak','kartik','nikunj']
    });
 });
 app.get('/loop2', function(request, response){
    response.render('loop2',
    {
        friends : ['ankit','brijesh','chintan','darshak','kartik','nikunj']
    });
 });
 app.get('/loop3', function(request, response){
    response.render('loop3',
    {
        person : {name : 'ajay',age : 36,height : 6.2,gender : 'Male'}
    });
 });
 app.get('/loop4', function(request, response){
    response.render('loop4');
 });
app.listen(5000);
console.log("we are ready...");
```



```
doctype html
html
  head
    title conditional statement
  body
    h1 loop in pug <hr />
    ol
      each name in friends
        li=name
```

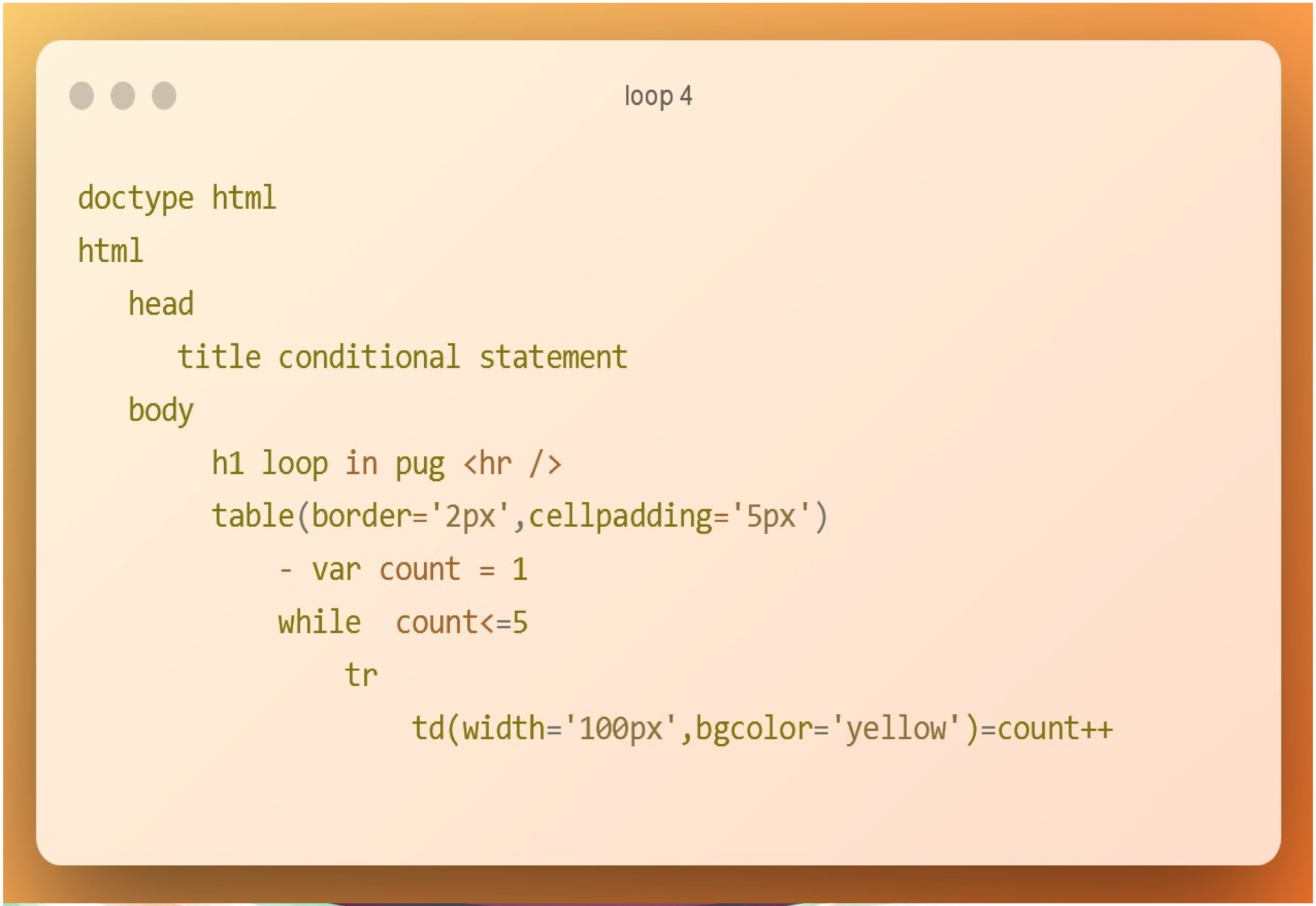


```
doctype html
html
   head
      title conditional statement
   body
        h1 loop in pug <hr />
        table(border='2px',cellpadding='5px')
            each name,index in friends
                tr
                    td(width='50px')=index+1
                    td(width='200px')=name
```



loop 3

```
doctype html
html
   head
      title conditional statement
   body
        h1 loop in pug <hr />
        table(border='2px',cellpadding='5px')
            each value,key in person
                tr
                    td(width='100px',bgcolor='grey')=value
                    td(width='200px')=key
```

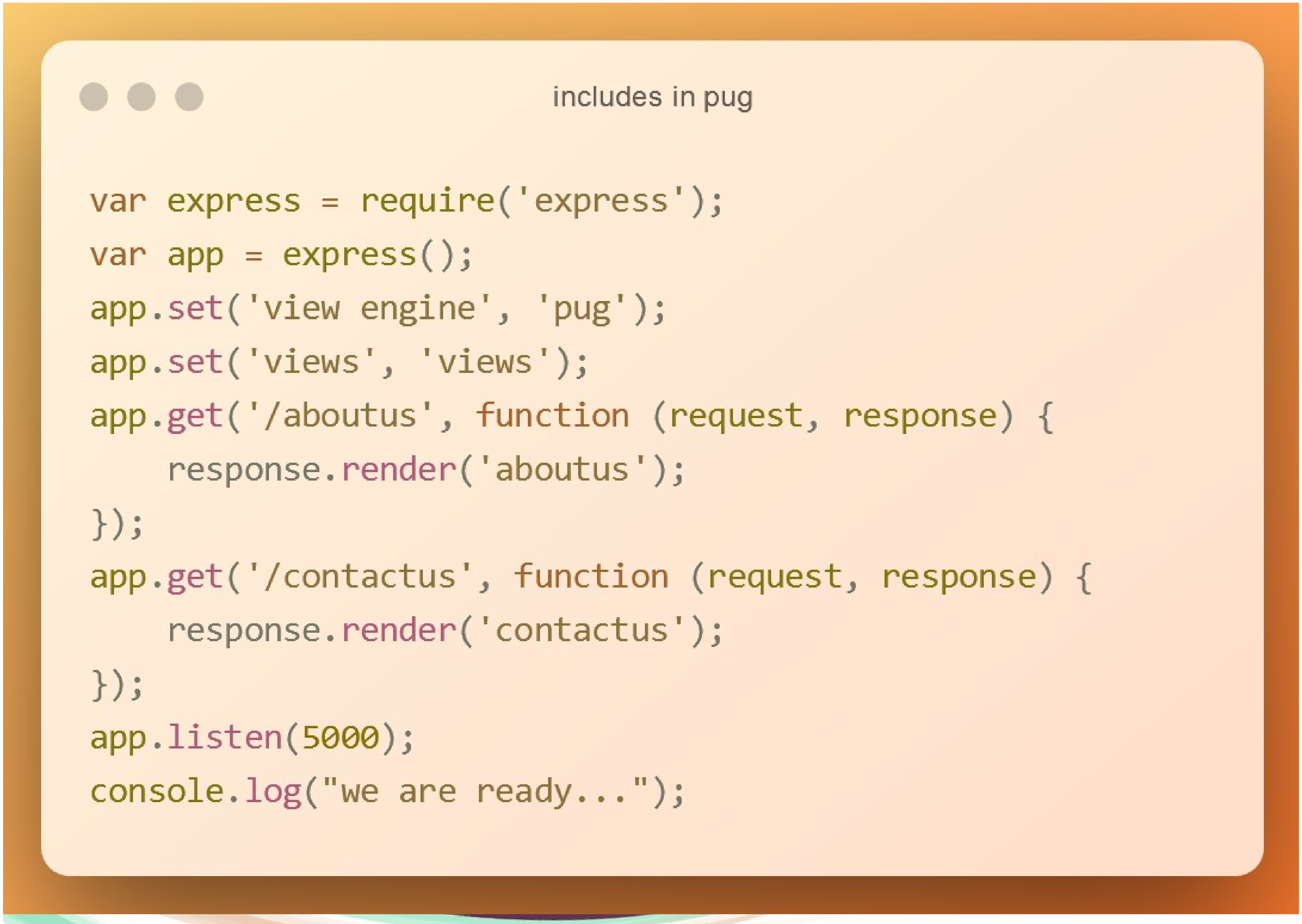


loop 4

```
doctype html
html
   head
      title conditional statement
   body
        h1 loop in pug <hr />
        table(border='2px',cellpadding='5px')
            - var count = 1
            while  count<=5
                tr
                    td(width='100px',bgcolor='yellow')=count++
```

Include in pug

- It is sometimes required to include one pug file into another pug file.
- We can do so using **includes**
- This help us to avoid writing same code again and again and update operations also be quick and easy.
- We should use it for header, footer and menus of the site because these components are common in each file.



includes in pug

```
var express = require('express');
var app = express();
app.set('view engine', 'pug');
app.set('views', 'views');
app.get('/aboutus', function (request, response) {
    response.render('aboutus');
});
app.get('/contactus', function (request, response) {
    response.render('contactus');
});
app.listen(5000);
console.log("we are ready...");
```

- header pug

```
head
  title this is header pug
```

- footer pug

```
footer#footer
  p(align='center') Copyright (c) the easylearn academy
```

- aboutus pug

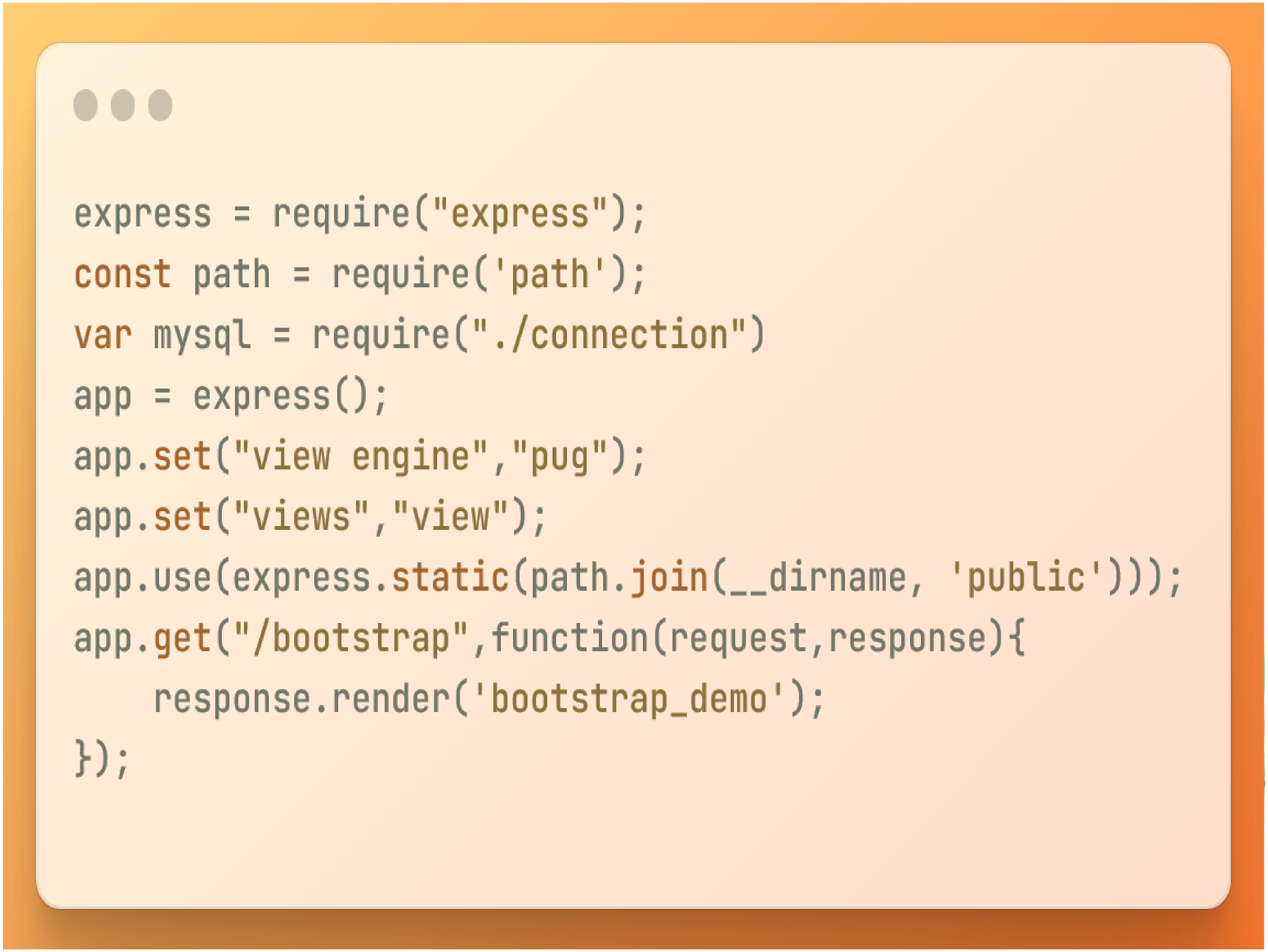
```
doctype html
html
  include header.pug
  body
    h1 My Site
    p this is about us page content
    include footer.pug
```

- Contact us pug

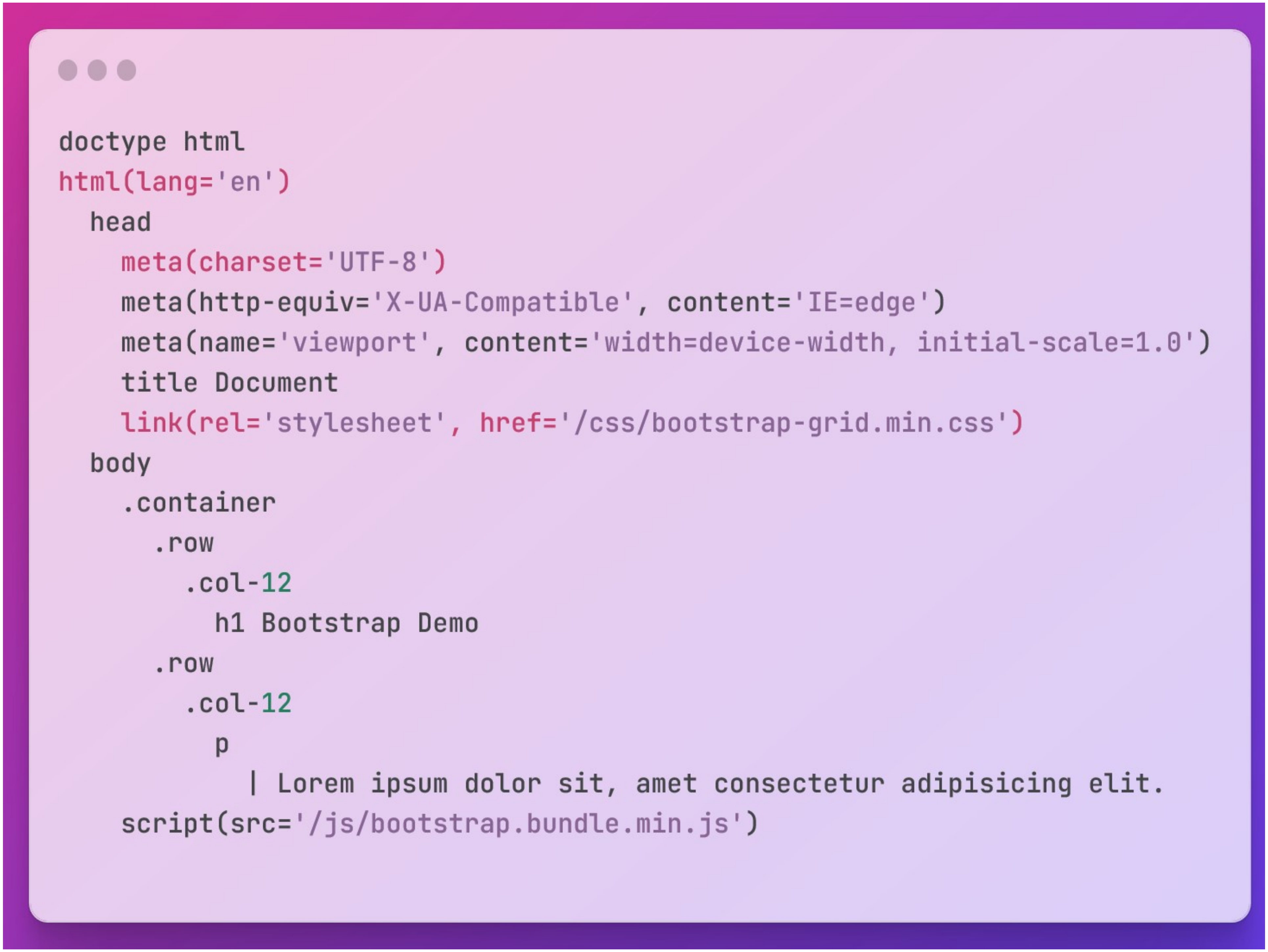
```
doctype html
html
  include header.pug
  body
    h1 My Site
    p this is contact us page content
    include footer.pug
```

How to use local css/javascript into pug file?

- It is possible to load css/javascript from project folder.
- To do that first create public folder in same folder which contain .js file which render the pug file.
- And then create css folder in public folder and put css file into this folder.
- And then create js folder in public folder and put js file into js folder.
- Now here we assume that css folder has bootstrap.min. Css file while js folder has bootstrap.min bootstrap.bundle.min.js file.
- Now use path module in .js file and use app.use middleware to set path.
- For better understanding refer code in next slide

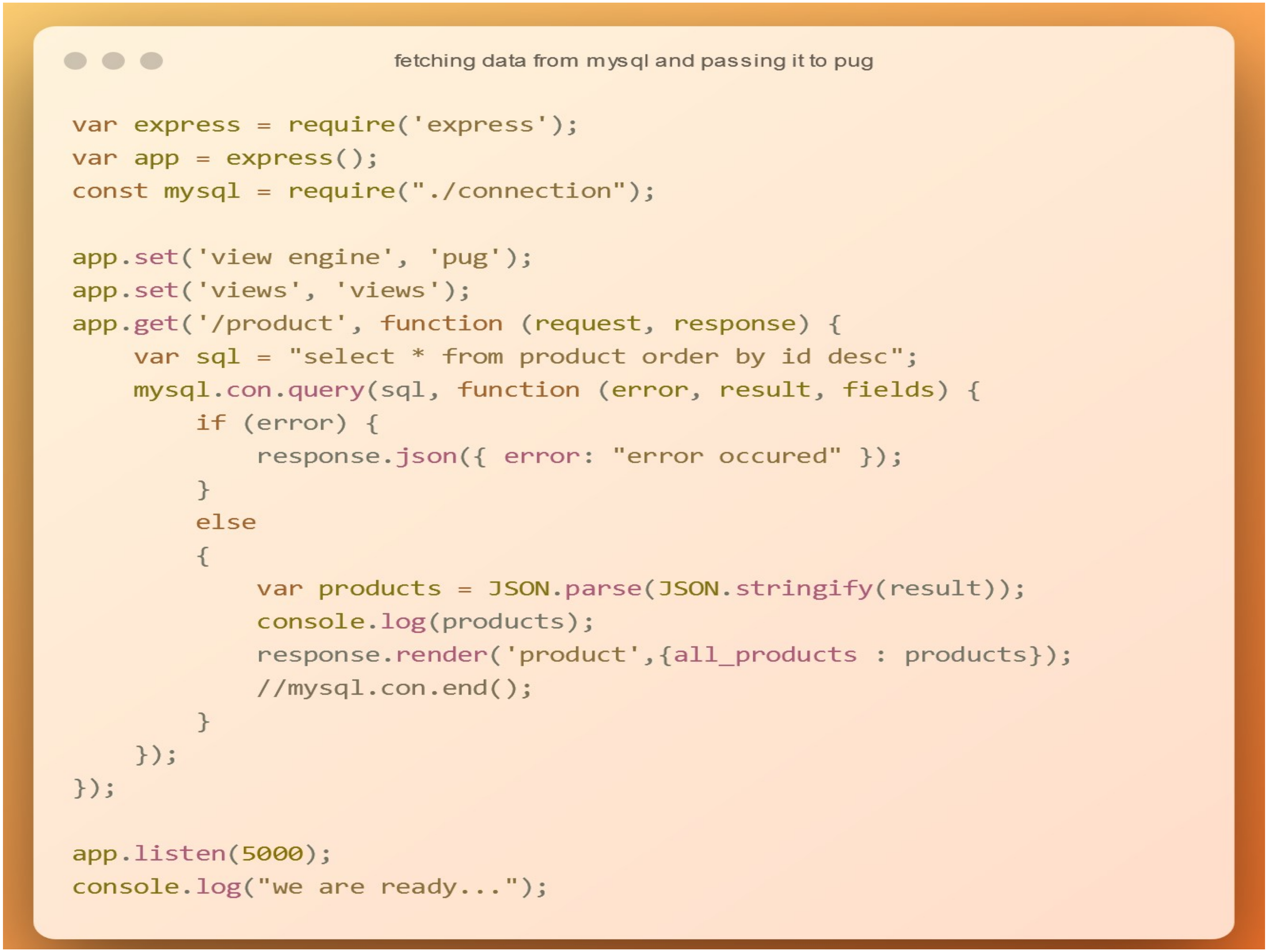


```
express = require("express");
const path = require('path');
var mysql = require("./connection")
app = express();
app.set("view engine","pug");
app.set("views","view");
app.use(express.static(path.join(__dirname, 'public')));
app.get("/bootstrap",function(request,response){
    response.render('bootstrap_demo');
});
```



```
doctype html
html(lang='en')
  head
    meta(charset='UTF-8')
    meta(http-equiv='X-UA-Compatible', content='IE=edge')
    meta(name='viewport', content='width=device-width, initial-scale=1.0')
    title Document
    link(rel='stylesheet', href='/css/bootstrap-grid.min.css')
  body
    .container
      .row
        .col-12
          h1 Bootstrap Demo
      .row
        .col-12
          p
            | Lorem ipsum dolor sit, amet consectetur adipisicing elit.
    script(src='/js/bootstrap.bundle.min.js')
```

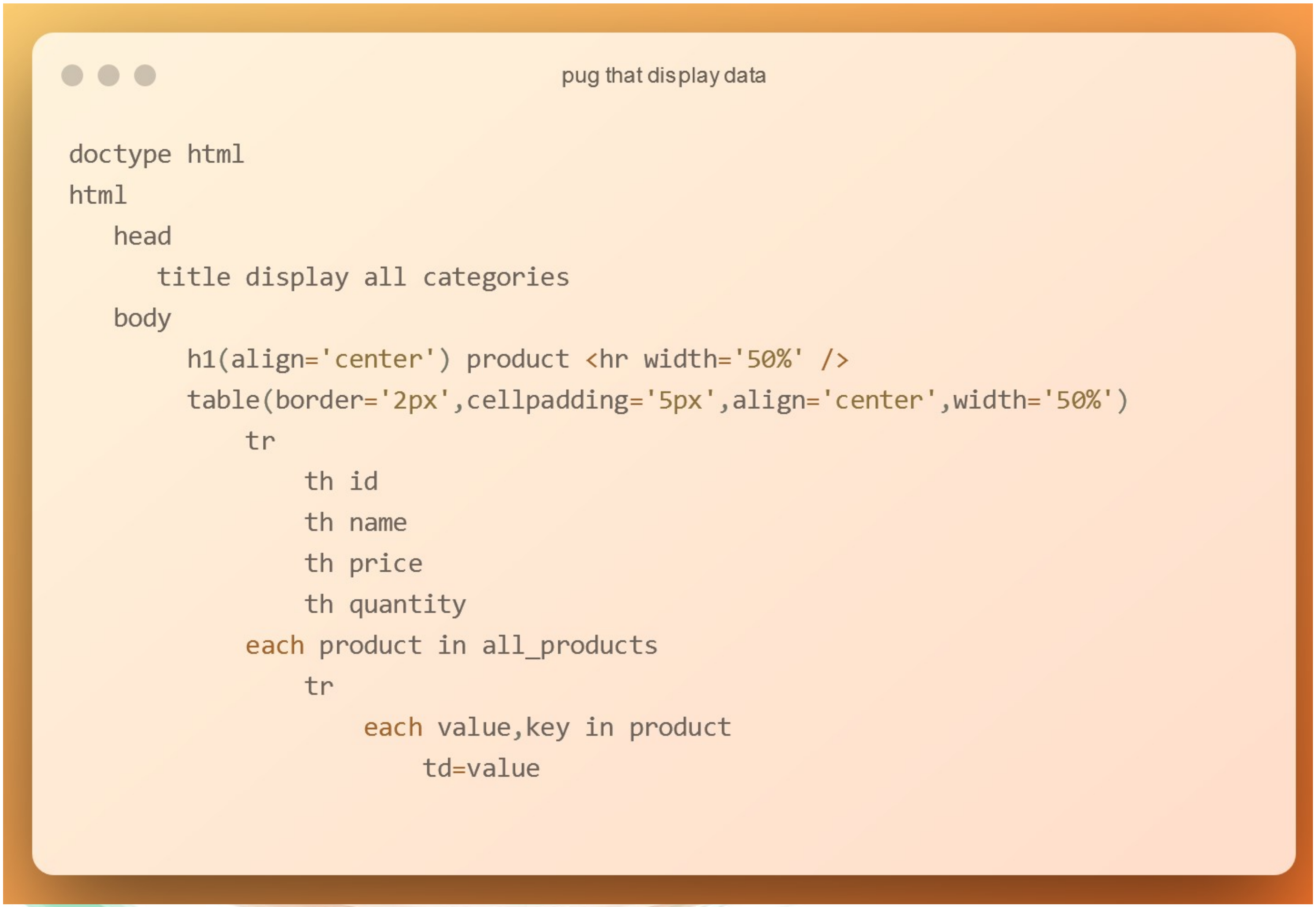
HOW TO DISPLAY DYNAMIC DATA USING NODEJS AND PUG


fetching data from mysql and passing it to pug

```
var express = require('express');
var app = express();
const mysql = require("./connection");

app.set('view engine', 'pug');
app.set('views', 'views');
app.get('/product', function (request, response) {
    var sql = "select * from product order by id desc";
    mysql.con.query(sql, function (error, result, fields) {
        if (error) {
            response.json({ error: "error occured" });
        }
        else
        {
            var products = JSON.parse(JSON.stringify(result));
            console.log(products);
            response.render('product',{all_products : products});
            //mysql.con.end();
        }
    });
});

app.listen(5000);
console.log("we are ready...");
```



pug that display data

```
doctype html
html
   head
      title display all categories
   body
        h1(align='center') product <hr width='50%' />
        table(border='2px',cellpadding='5px',align='center',width='50%')
            tr
                th id
                th name
                th price
                th quantity
            each product in all_products
                tr
                    each value,key in product
                        td=value
```