

The background is a child's painting. The top half features a dense canopy of trees with thick, textured brushstrokes in shades of blue and purple. The bottom half shows a green field with several thin, dark tree trunks. In the distance, there is a house with a red roof and a red arched doorway. A small figure of a person is visible near the base of one of the trees in the foreground.

Modern FrontEnd Devops

104 兒童美術館自動部署分享

Kenny Lee



Kenny Lee

平台研發 Team Leader / Devops 專案經理 /
cicisasa 架構規劃開發

網站軟體架構的狂熱分子，擅長各項新技術的研發導入

Email : microbean@gmail.com

Github : <https://github.com/KennyTw>

104+ : <http://plus.104.com.tw/>李坤承



Today Topic

FrontEnd Devops concept

cicisasa tech stack

cicisasa deployment procedure

QA



Why FrontEnd **Devops ?**



由於前端開發技術越來越複雜


So many Js & CSS




但是許多 Ops 偏向 後端 技術

他們不懂

為何前端如此複雜



所以我們需要一個好的架構
及上線規劃



而且我們也需要了解一些後端技術



FrontEnd **Devops** Plan & Goal

The background features a light blue gradient with two large, overlapping semi-circular shapes at the top. Below these, there are four stylized human figures in a light blue color. The second figure from the left is holding a tablet displaying a webpage. The third figure is holding a tablet displaying a network diagram with nodes and connecting lines. The text "使用自動化" is centered over the figures in a large, white, sans-serif font.

使用自動化

降低人工操作的**成本**

The background features a light blue gradient with two large, semi-transparent circular shapes. Inside these circles are faint icons: on the left, a person sitting at a computer with a webpage on the screen; on the right, a person sitting at a computer with a network diagram on the screen. The text '需要有好的 JS/CSS 框架' is centered in the lower half of the image.

需要有好的 JS/CSS 框架



利用HotDeploy
實現

Zero Downtime



實現隨時都可以上線

ERROR

500 error
需要被 handle



兒童美術館

cicisasa 是如何做到呢？

cicisasa.com

免費儲存孩子圖畫的網路空間
保存孩子的童年回憶及創意

技術實驗場

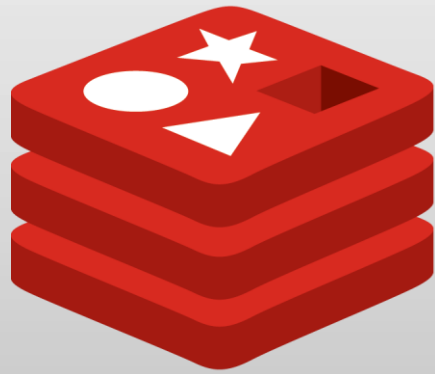


Code language



Light, Easy, Rich ecosystem

Database



redis

Light, High Performance

FrontEnd Framework



simple and quickly

JS Framework



Backbone.js

Light,simple

Task manager



rich ecosystem

Dependency manager



`require('modules')` is simple

Continuous Integration



Travis CI

most popular

Perfectly integrates with GitHub

Template Engine



Both Client/Server render

Test Framework



supertest , should

Love "TJ Holowaychuk" (ejs..)

Client/Server



socket.io

Avoid too many ajax request



Cicisasa HotDeploy “naught”

naught • Zero downtime deployment project

- **Node cluster** : Clustering in Node.js allows you to create separate processes which can share same server port

- **naught start** --ipc-file **server.ipc**
--stdout stdout.log
--stderr stderr.log ./apps/server/**server.js**
- **naught deploy server.ipc**

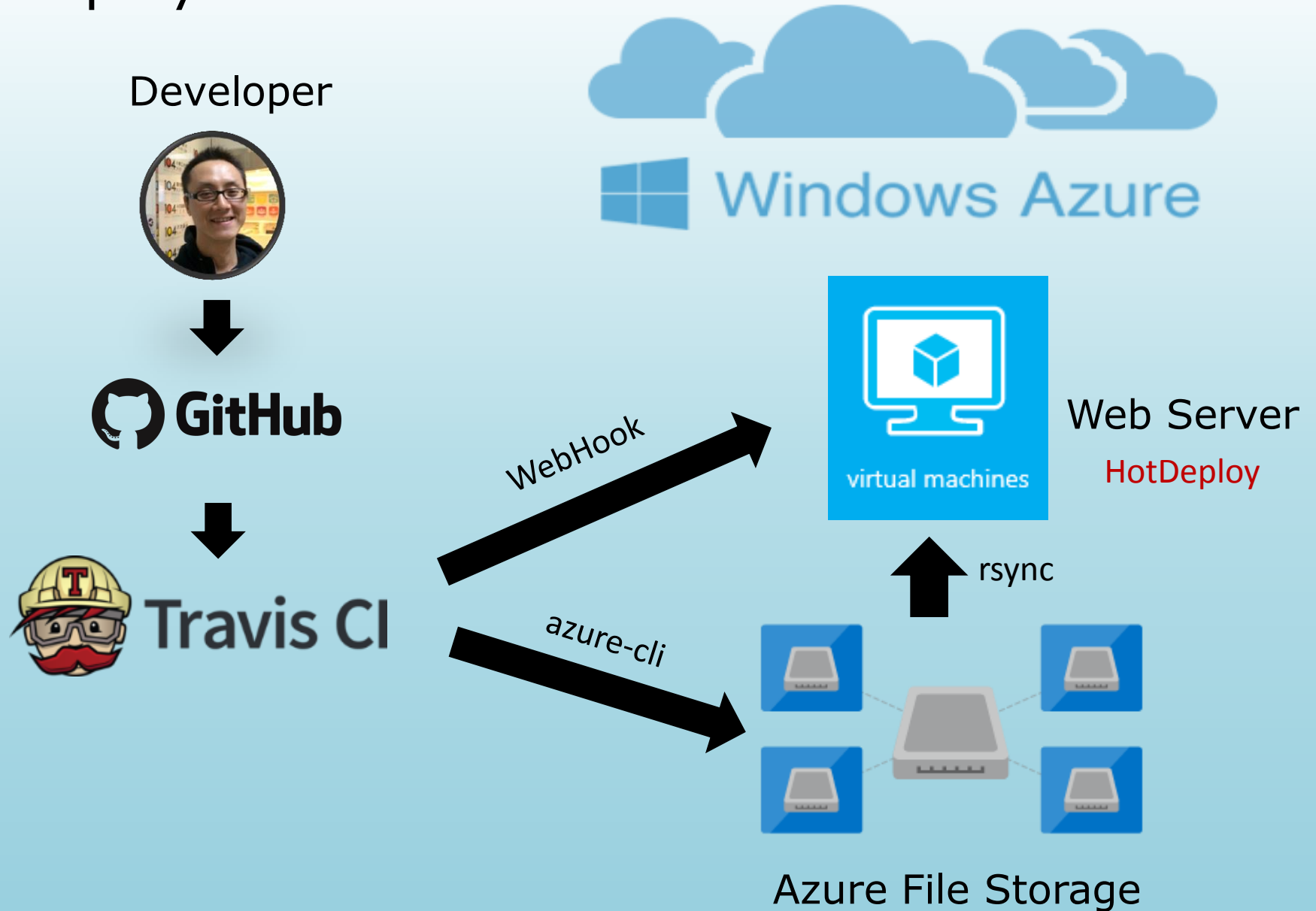
server.js

```
var domain = require('domain') ,  
serverDomain = domain.create() ,  
gracefulExit = require('express-graceful-exit');  
serverDomain.run(function() {  
  var server = http.createServer(app);  
  process.on('message', function(message) {  
    if (message === 'shutdown')  
      { gracefulExit.gracefulExitHandler(app, server, {log: true,  
        suicideTimeout: 6*1000});
```

The background features a light blue gradient with two large, overlapping semi-circles at the top. Below these, there are faint, stylized icons representing a DevOps workflow: a person on the left, a monitor displaying a web page in the center-left, a monitor displaying a network diagram in the center-right, and another person on the right. The main title is centered in a bold, black font.

Cicisasa Deploy Procedure

Deploy Procedure





GitHub

Code Commit



Travis CI

- **CI initial**
(Npm install , redis-db , test data)
- **JS/css transform& copy & pack & uglify/minify**
- **Test**
- **Source Code transfer to Azure**
- **HotDeploy**



Next



Travis CI CI initial (.travis.yml)

services:

- redis-server

before_install:

- npm install -g grunt-cli
- npm install -g bower@1.4.1
- npm install -g azure-cli
- npm install
- node travis_ini.js

```
var redis = require('redis');  
var db = redis.createClient();  
db.set("data","helloworld");
```



Travis CI CI initial (.travis.yml)

- `chmod -R 777 ./travis_notify.sh`
- `chmod -R 777 ./travis_grunt.sh`
- `chmod -R 777 ./travis_azure.sh`
- `chmod -R 777 ./app`
- `bower install`



Travis CI

JS/css transform & copy & pack & uglify

(.travis.yml)

- ./travis_grunt.sh

```
if [ "$TRAVIS_BRANCH" == "master" ]; then
    grunt ci
    grunt test
fi
```

Gruntfile.js

```
grunt.registerTask('ci', ['shell:start', 'shell:bower', 'shell:npm', 'concat', 'cssmin', 'browserify:prod', 'uglify', 'shell:deploy']);
```




Travis CI Test

Gruntfile.js

```
grunt.registerTask('test', ['mochaTest:test']);
```

```
mochaTest: { test: { src: ['test/*.js'] }}
```

/test/RootBasicTest.js (ServerSide Test)

```
var request = require('supertest');  
var should = require('should');  
describe('Basic Server Test',  
  function() { this.timeout(10000); //timeout with 10 secs  
    it('Get / should respond 200 and check some keyword',  
      function(done) {  
        request("http://127.0.0.1:3000")  
          .get('/')  
          .expect(200, done)  
          .end(function(err, res) {  
            res.text.should.containEql('/js/bundle.js');  
            done(err);  
          });  
      });  
  });
```



Travis CI Test

/test/RootJsTest.js (Client Side Test)

```
var request = require('supertest');  
var should = require('should');  
var phantom = require('phantom');  
  
describe('Basic Client Test with PhantomJS', function() {  
  var page;  
  ...  
  it('get / should return data back and no js error',  
    function (done) {  
    page.set('onError', function(error) { done(error); });  
    page.open('http://127.0.0.1:3000', function (status) {  
      page.evaluate(  
        function () { return document.title; } ,  
        function (result)  
        if (result.length > 0) { done(); } else  
        { done("document.title null"); }  
      );  
    });  
  });  
});
```



Travis CI SourceCode transfer to Azure

- ./travis_azure.sh

```
echo "tar files..."
```

```
tar -zcvf "./source.tgz" "./"
```



```
echo "Uploading files..."
```

```
azure storage file upload -q "./source.tgz" deploy
```





Travis CI HotDeploy

- ./travis_notify.sh

```
if [ "$TRAVIS_BRANCH" == "master" ]; then
echo "master deploy"
curl --data ""
http://ithomefedevops.cloudapp.net:3000/travis_web_hook
fi
```



HotDeploy

server.js

```
app.post('/travis_web_hook', function(req, res) {  
  spawn('grunt', [], {cwd: '/home/azureuser/code'});  
  res.end();  
});
```

Gruntfile.js

```
grunt.registerTask('default', ['shell:tar',  
'shell:rsync', 'shell:npm', 'shell:deploy']);
```





Resource Here

<https://github.com/KennyTw/ITHomeFEDevops>



QA



Q:專案規模到何種程度的時候
需要安裝自動部署



Q: Javascript Unit Test與
End to End Test的重要性，
是否都得進行？



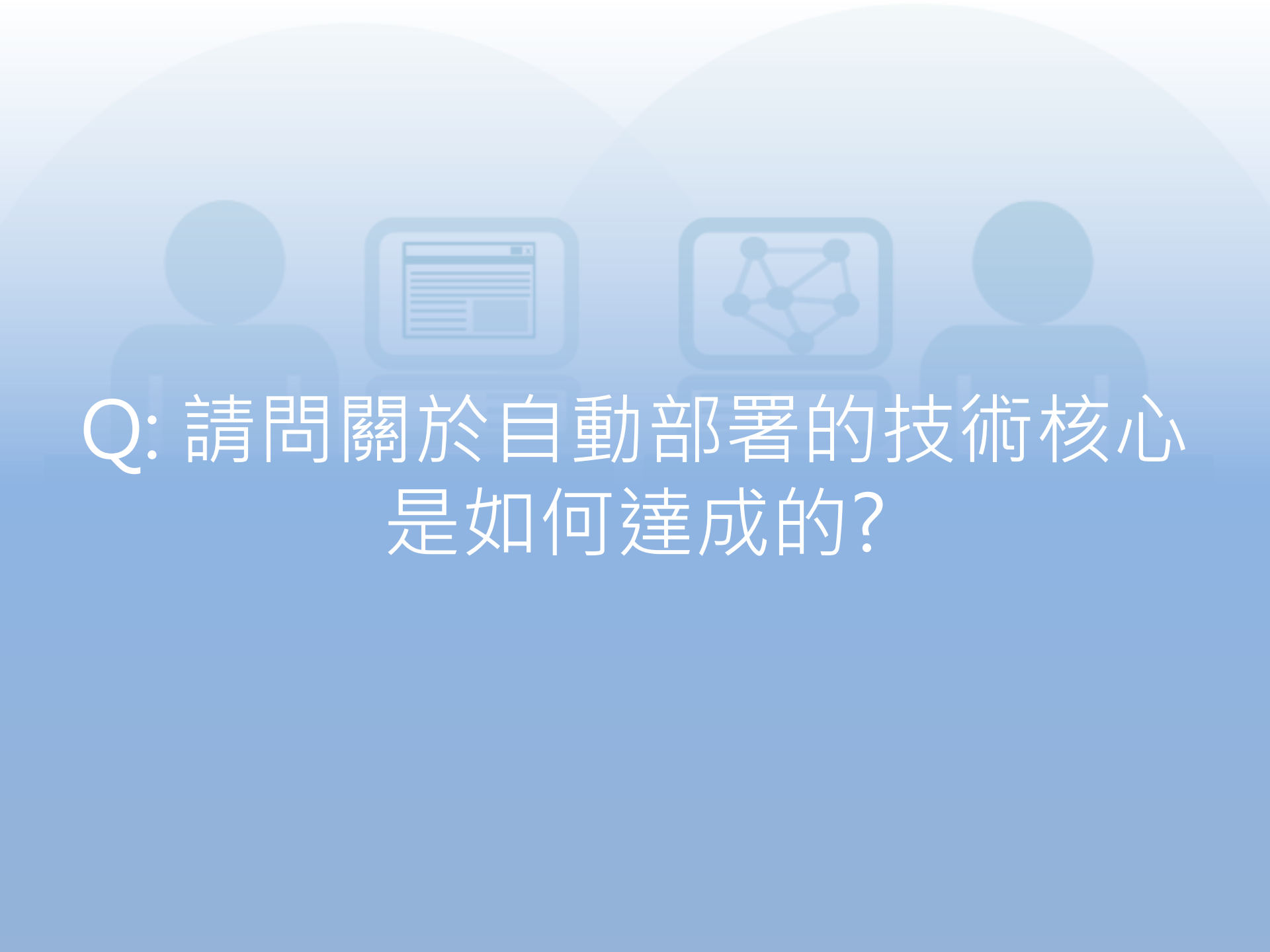
Q: 自動部屬的學習門檻高嗎?



Q: 自動部署の信頼性



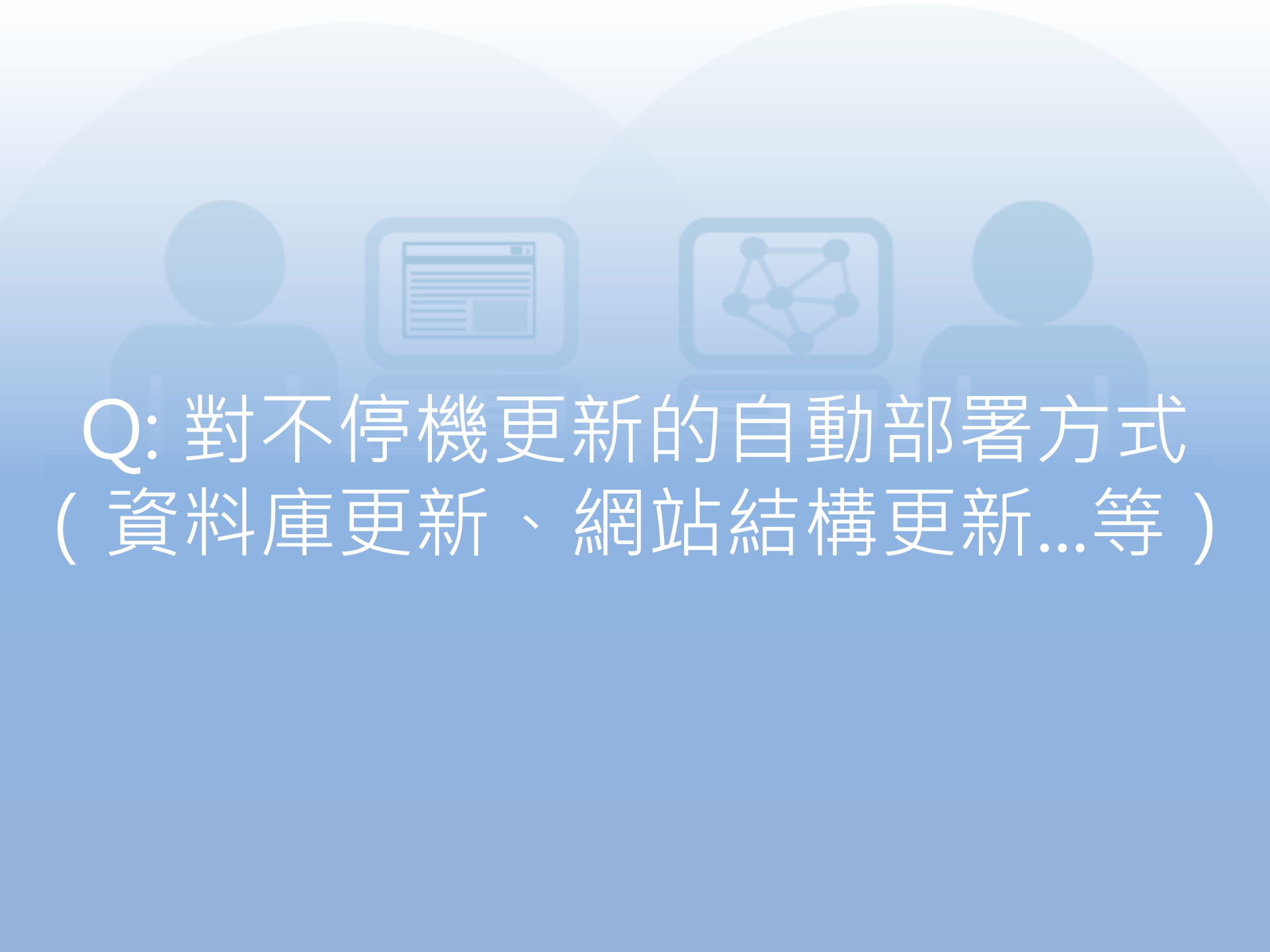
Q: 是否透過git hook自動deploy?



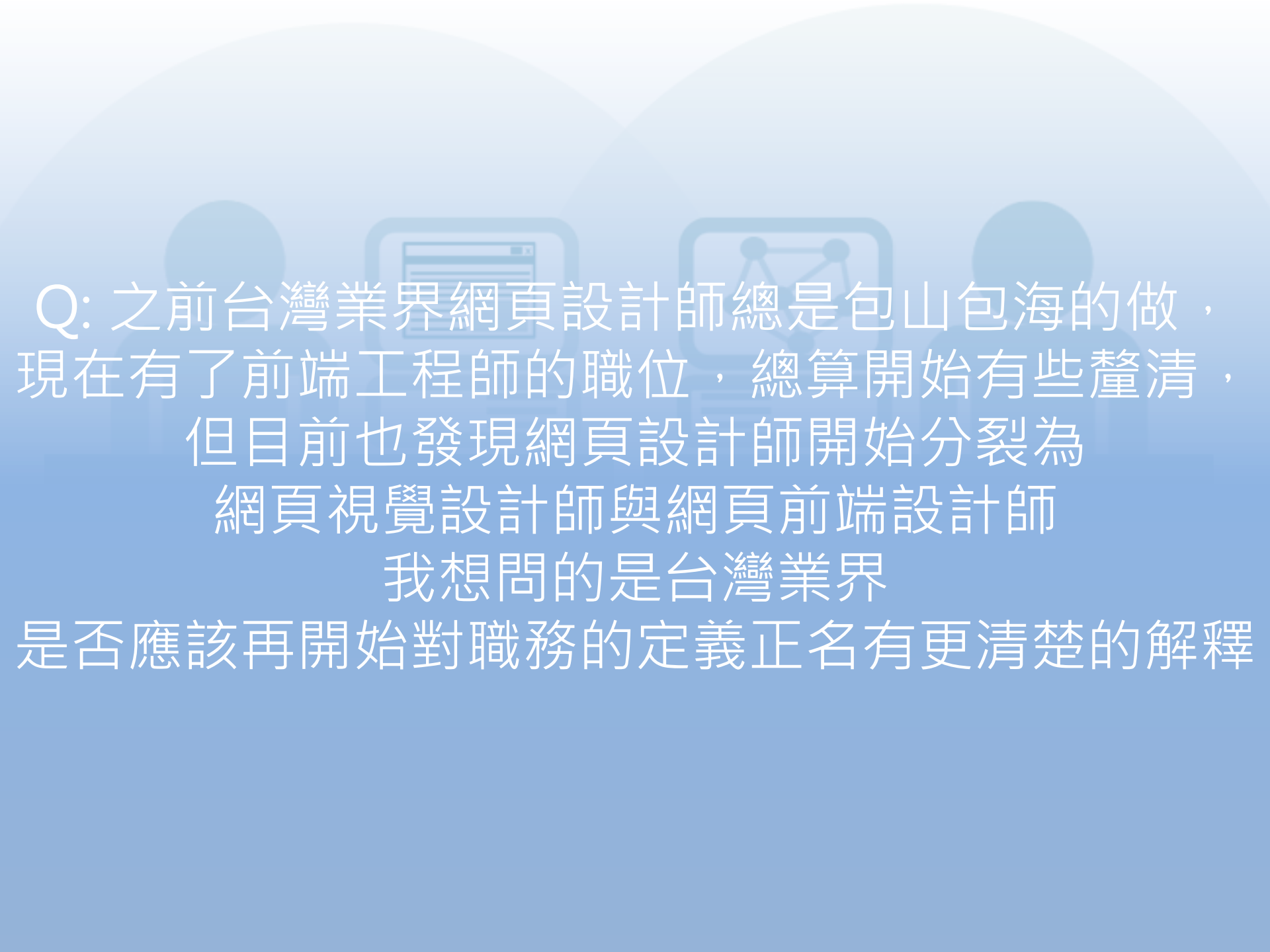
Q: 請問關於自動部署的技術核心
是如何達成的？



Q: 適用於雲端主機嗎?
若有多台主機,可以一次搞定嗎?



Q: 對不停機更新的自動部署方式
(資料庫更新、網站結構更新...等)



Q: 之前台灣業界網頁設計師總是包山包海的做，
現在有了前端工程師的職位，總算開始有些釐清，
但目前也發現網頁設計師開始分裂為
網頁視覺設計師與網頁前端設計師
我想問的是台灣業界
是否應該再開始對職務的定義正名有更清楚的解釋

The background features a blue gradient with faint, semi-transparent icons. On the left and right are stylized human figures. In the center, there are two computer monitors; the left one shows a webpage layout, and the right one displays a network diagram with interconnected nodes. Above these are two large, overlapping light-blue circular shapes.

Thank You