4 Methods for Writing Multi-Threaded Code in Java

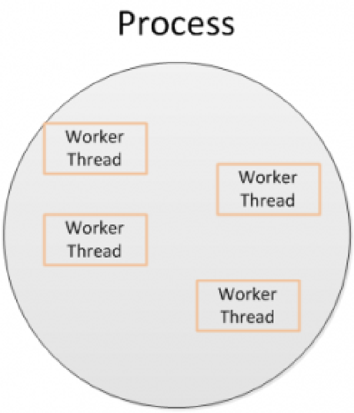
By Jay Sridhar / July 5, 2017 05-07-2017 / 6 minutes

Jay Sridhar
2 articles

Facebook Twitter Pinterest
Stumbleupon Email

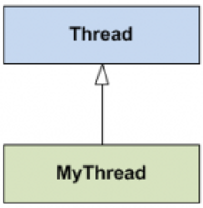
Multi-threading is a method of writing code for executing tasks in parallel. Java has had excellent support for writing multi-threaded code since the early days of Java 1.0. Recent enhancements to Java have increased the ways in which code can be structured to incorporate multi-threading in Java programs.

In this article, we compare a few of these options so you can better judge which option to use for your next Java project.



Method 1: Extending the Thread class

ask. When you want to kick off the task in its own thread, you can create an instance of this class and invoke its **start()** method. This starts the thread execution and runs to completion (or terminates in an exception).



Here is a simple Thread class which just sleeps for a specified interval as a way of simulating a long-running operation.

```
public class MyThread extends Thread
{
  private int sleepFor;

  public MyThread(int sleepFor) {
    this.sleepFor = sleepFor;
  }

  @Override
  public void run() {
    System.out.printf("[%s] thread starting\n",
    Thread.currentThread().toString());
    try { Thread.sleep(this.sleepFor); }
    catch(InterruptedException ex) {}
    System.out.printf("[%s] thread ending\n",
    Thread.currentThread().toString());
  }
}
```

Create an instance of this Thread class by giving it the number of milliseconds to sleep.

Kick off the execution of this worker thread by invoking its start() method. This method returns control immediately to the caller, without waiting for the thread to terminate.

```
worker.start();
System.out.printf("[%s] main thread\n", Thread.cu
```

And here is the output from running this code. It indicates that the main thread diagnostic is printed before the worker thread executes.

```
[Thread[main,5,main]] main thread
[Thread[Thread-0,5,main]] thread starting
[Thread[Thread-0,5,main]] thread ending
```

Since there are no more statements after starting the worker thread, the main thread waits for the worker thread to finish before the program exits. This allows the worker thread to complete its task.

Method 2: Using a Thread Instance With a Runnable

Java also provides an interface called **Runnable** which can be implemented by a worker class to execute the task in its **run()** method. This is an alternative way of creating a worker class as opposed to extending the **Thread** class (described above).

Related Articles

PROGRAMMING

Java vs. JavaScript: Everything You Need to Know

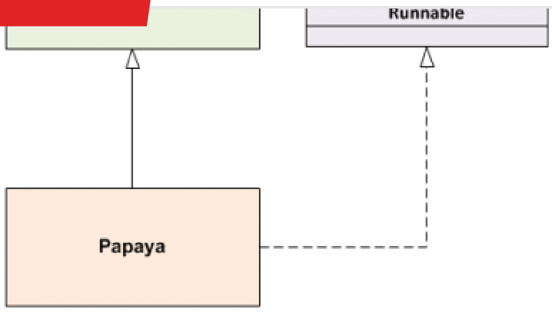
PROGRAMMING

How to Handle Java Exceptions the Right Way

WINDOWS , SECURITY

4 Windows Apps to Keep Updated at All Times

Affiliate Disclosure: By buying the products we recommend, you help keep the lights on at MakeUseOf. .



Here is the implementation of the worker class which now implements Runnable instead of extending Thread.

```
public class MyThread2 implements Runnable {
  // same as above
}
```

The advantage of implementing the Runnable interface instead of extending the Thread class is that the worker class can now extend a domain-specific class within a class hierarchy.

What does this mean?

Let us say, for example, you have a **Fruit** class which implements certain generic characteristics of fruits. Now you want to implement a **Papaya** class which specializes certain fruit characteristics. You can do that by having the **Papaya** class extend the **Fruit** class.

```
public class Fruit {
  // fruit specifics here
}

public class Papaya extends Fruit {
  // override behavior specific to papaya here
}
```

Now suppose you have some time-consuming task that Papaya needs to support, which can be performed in a separate thread. This case can be handled by having the

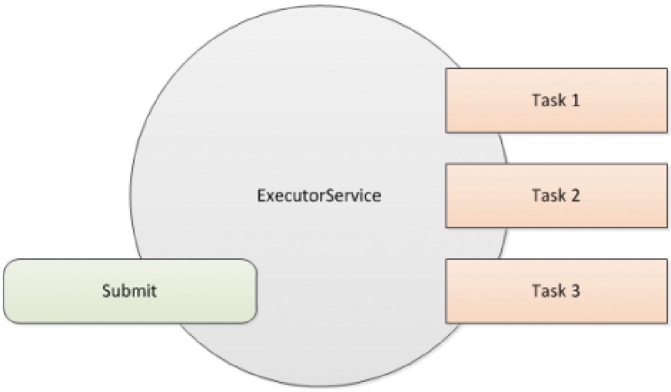
```
public class Papaya extends Fruit implements Runı
  // override behavior specific to papaya here

  @Override
  public void run() {
    // time consuming task here.
  }
}
```

To kick off the worker thread, you create an instance of the worker class and hand it over to a Thread instance at creation. When the start() method of the Thread is invoked, the task executes in a separate thread.

```
Papaya papaya = new Papaya();
// set properties and invoke papaya methods here
Thread thread = new Thread(papaya);
thread.start();
```

And that is a brief summary of how to use a Runnable to implement a task executing within a thread.

Method 3: Execute a Runnable With ExecutorService



generalizes the concept of thread execution by abstracting away creation of threads.

This is because you can run your tasks within a pool of threads just as easily as using a separate thread for each task. This allows your program to track and manage how many threads are being used for worker tasks.

Suppose you have a 100 worker tasks waiting to be executed. If you start one thread per worker (as presented above), you would have 100 threads within your program which might lead to bottlenecks elsewhere within the program. Instead, if you use a thread pool with, say 10 threads pre-allocated, your 100 tasks will be executed by these threads one after another so your program is not starved for resources. In addition, these thread pool threads can be configured so that they hang around to perform additional tasks for you.

An ExecutorService accepts a **Runnable** task (explained above) and runs the task at a suitable time. The **submit()** method, which accepts the Runnable task, returns an instance of a class called **Future**, which allows the caller to track the status of the task. In particular, the **get()** method allows the caller to wait for the task to complete (and provides the return code, if any).

In the example below, we create an ExecutorService using the static method **newSingleThreadExecutor()**, which as the name indicates, creates a single thread for executing tasks. If more tasks are submitted while one task is running, the ExecutorService queues up these tasks for subsequent execution.

The Runnable implementation we use here is the same one described above.

```
ExecutorService esvc = Executors.newSingleThread
Runnable worker = new MyThread2(sleepFor);
Future<?> future = esvc.submit(worker);
System.out.printf("[%s] main thread\n", Thread.c
```

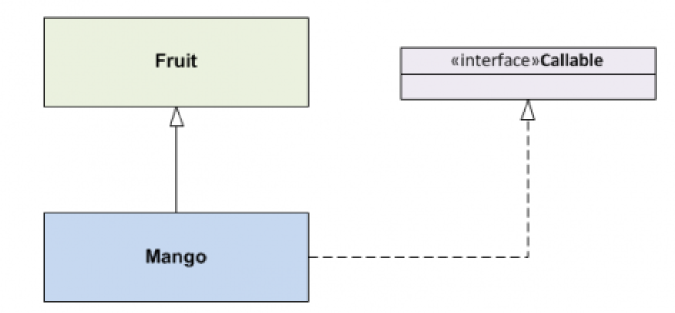
Note that an ExecutorService must be properly shut down when it is no longer needed for further task submissions.

Method 4: A Callable Used With ExecutorService

Starting with version 1.5, Java introduced a new interface called **Callable**. It is similar to the older Runnable interface with the difference that the execution method (called **call()** instead of **run()**) can return a value. In addition, it can also declare that an **Exception** can be thrown.

An ExecutorService can also accept tasks implemented as **Callable** and returns a **Future** with the value returned by the method at completion.

Here is an example **Mango** class which extends the **Fruit** class defined earlier and implements the **Callable** interface. An expensive and time-consuming task is performed within the **call()** method.



```
public class Mango extends Fruit implements Call
  public Integer call() {
    // expensive computation here
    return new Integer(0);
  }
}
```

prints its return value.

```
ExecutorService esvc = Executors.newSingleThread

MyCallable worker = new MyCallable(sleepFor);
Future future = esvc.submit(worker);
System.out.printf("[%s] main thread\n", Thread.c
System.out.println("Task returned: " + future.ge
esvc.shutdown();
```

What Do You Prefer?

In this article, we learned a few methods to write multi-threaded code in Java. These include:

1. Extending the **Thread** class is the most basic and has been available from Java 1.0.
2. If you have a class which must extend some other class in a class hierarchy, then you can implement the **Runnable** interface.
3. A more modern facility for creating threads is the **ExecutorService** which can accept a Runnable instance as a task to run. The advantage of this method is that you can use a thread pool for task execution. A thread pool helps in resource conservation by reusing threads.
4. Lastly, you can also create a task by implementing the **Callable** interface and submitting the task to an ExecutorService.

Which of these options do you think you will use in your next project? Let us know in the comments below.

0 COMMENTS

WRITE A COMMENT

PROGRAMMING

How to Install WordPress on a Virtual Server for Free Using Cloud9

By **Khamosh Pathak** / July 4, 2017 04-07-2017 / 2 minutes

Khamosh Pathak

41 articles

Khamosh Pathak is a freelance technology writer and User Experience Designer. When he's not helping people make the best of their current technology, he's helping clients design better apps and websites. In his free time, you'll find him watching comedy specials on Netflix and trying once again, to get through…

Getting a local WordPress server up and running is surprisingly complicated. If you're a seasoned developer, it makes sense to set up a local server with **XAMPP or WAMP**. But if you're a beginner who just wants to mess around with a self-hosted WordPress installation (which is different from wordpress.com), to see what a customized WordPress site would look like, you're usually out of luck.

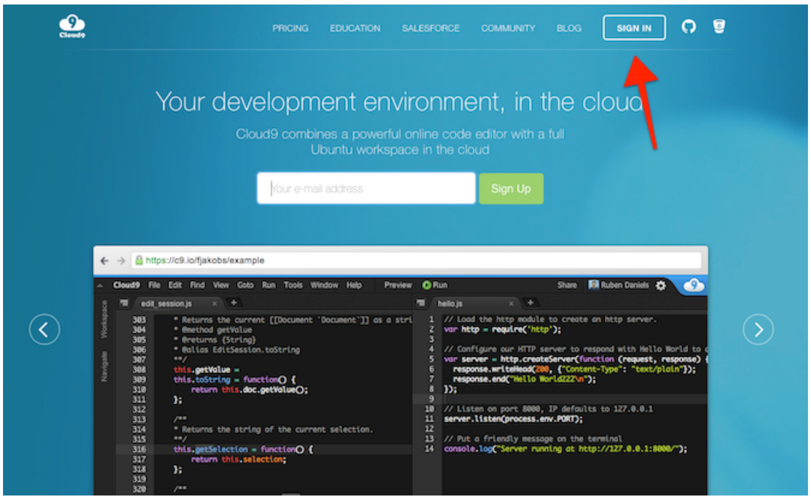
So I recommend you skip this tedious process. Instead, create a free virtual WordPress installation on a Cloud9 server space. There's no coding required. No need to download specialized software. You'll be tinkering with a WordPress installation — **customizing themes** and **adding plugins** — in just a couple of minutes.

What Is Cloud9?

Cloud9 is a cloud-based development environment. It lets you code in more than 40 languages on servers hosted by hard drive space, which is more than enough for our endeavor. Plus, installing WordPress on a workspace is just a one-click affair.

How to Install WordPress on Cloud9

Step 1: Visit Cloud9's website and click on **Sign in** at the top toolbar.



Your development environment, in the cloud

Cloud9 combines a powerful online code editor with a full Ubuntu workspace in the cloud

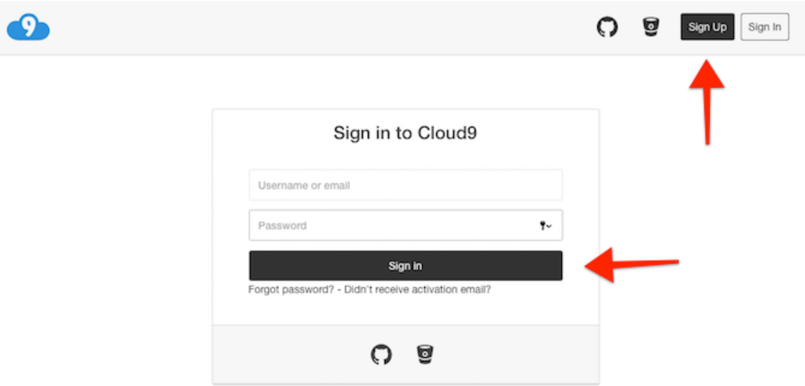
Your e-mail address

https://c9.io/fjakobs/example

```
    * Returns the current [[Document `Document`]] as a stri
    * @method getValue
    * @returns {String}
    * @alias EditSession.toString
    **/
    this.getValue =
    this.toString = function() {
        return this.doc.getValue();
    };

    /**
    * Returns the string of the current selection.
    **/
    this.getSelection = function() {
        return this.selection;
    };

    /**
```

```
// Load the http module to create an http server.
var http = require('http');

// Configure our HTTP server to respond with Hello World to 
var server = http.createServer(function (request, response) {
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.end("Hello World222\n");
});

// Listen on port 8000, IP defaults to 127.0.0.1
server.listen(process.env.PORT);

// Put a friendly message on the terminal
console.log("Server running at http://127.0.0.1:8000/");
```

Step 2: On the next page, sign up.



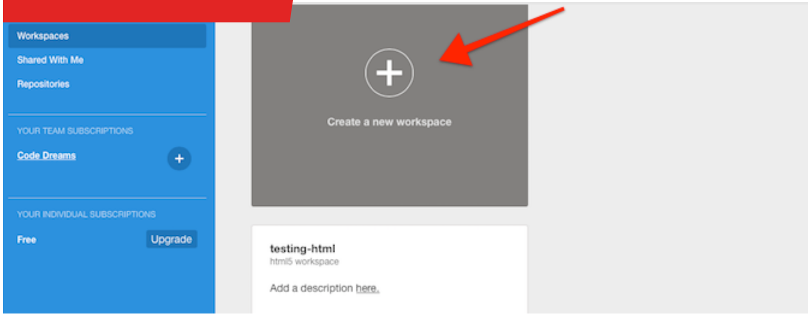
Sign in to Cloud9

Username or email

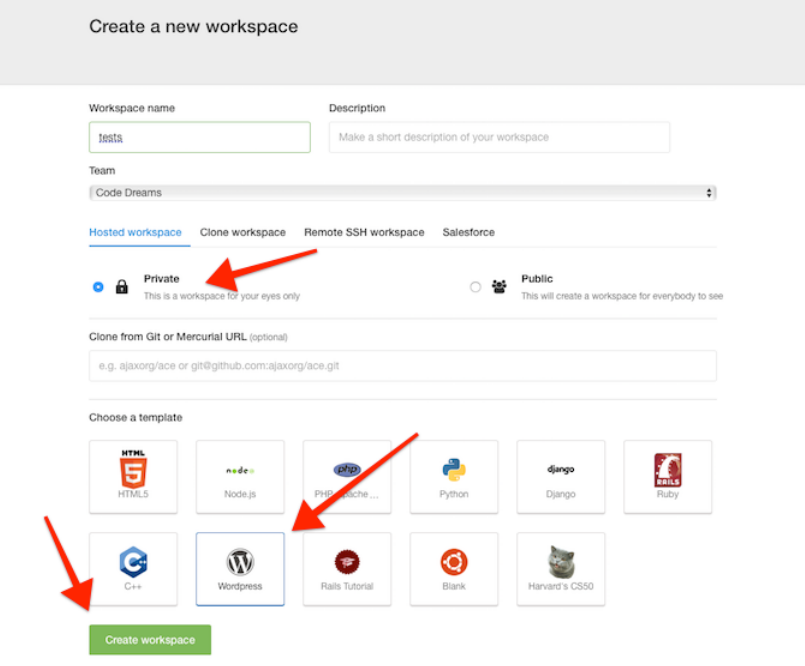
Password

Forgot password? - Didn't receive activation email?

Step 3: Once you're signed up, you'll see your **Workspaces** on your account page. Click on **Create a new workspace**.



Step 4: Give your workspace a name and select the team (if any). You can make a workspace **Private** (for free, you can only do this once). Now in the template section, click on WordPress. Then select **Create Workspace**.


Create a new workspace

Workspace name: tests

Description: Make a short description of your workspace

Team: Code Dreams

Hosted workspace

● Private
This is a workspace for your eyes only

○ Public
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional)

e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Choose a template: Wordpress

Step 5: Wait while Cloud9 whips up a new virtual server out of thin air. Take this time to appreciate the wonders of technology.

Trending

TECHNOLOGY EXPLAINED

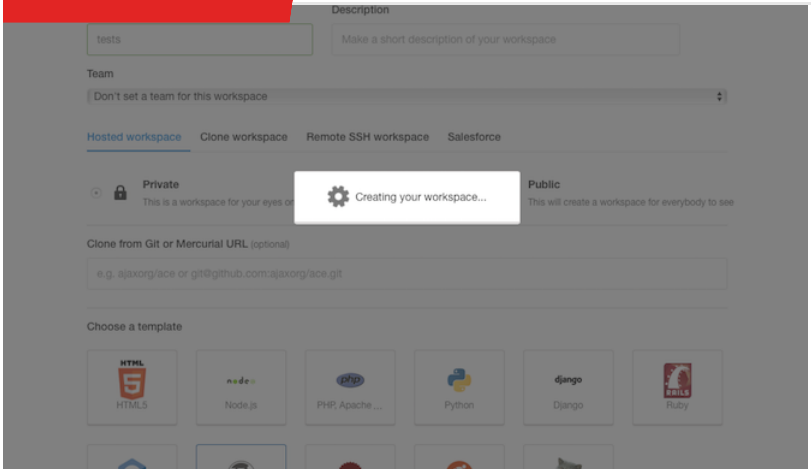
This Is Why Your Phone Slows Down Over Time

INTERNET , ENTERTAINMENT

Get an Instant $10 Amazon Credit Just by Streaming a…

INTERNET

How to Get 2% Off of All Amazon Prime Purchases…



Description: Make a short description of your workspace

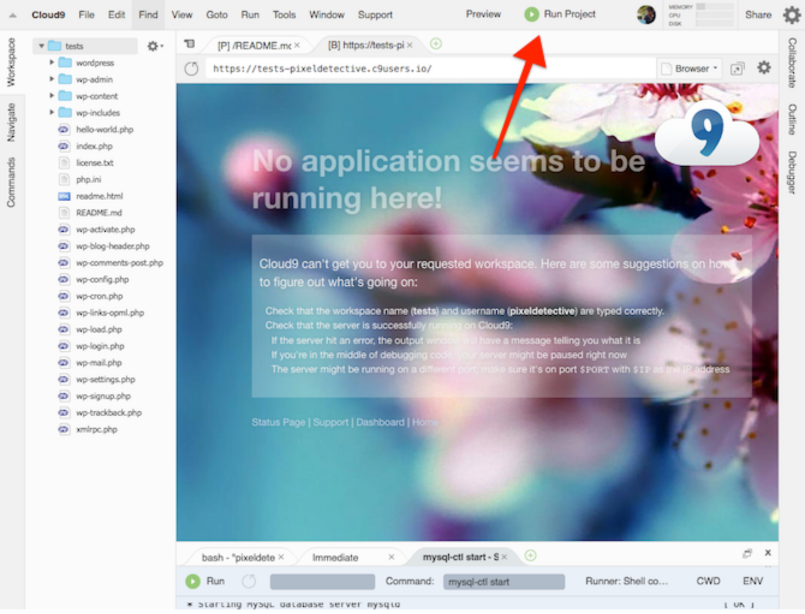
Name: tests

Team: Don't set a team for this workspace

Hosted workspace
Clone workspace
Remote SSH workspace
Salesforce
Private
This is a workspace for your eyes on
Creating your workspace...
Public
This will create a workspace for everybody to see

Clone from Git or Mercurial URL (optional): e.g. ajaxorg/ace or git@github.com:ajaxorg/ace.git

Step 6: You'll now land in what's called an IDE. Feel free to ignore all the code. Just click on the **Run Project** from the top toolbar.



https://tests-pixeldetective.c9users.io/

No application seems to be running here!

Cloud9 can't get you to your requested workspace. Here are some suggestions on how to figure out what's going on:

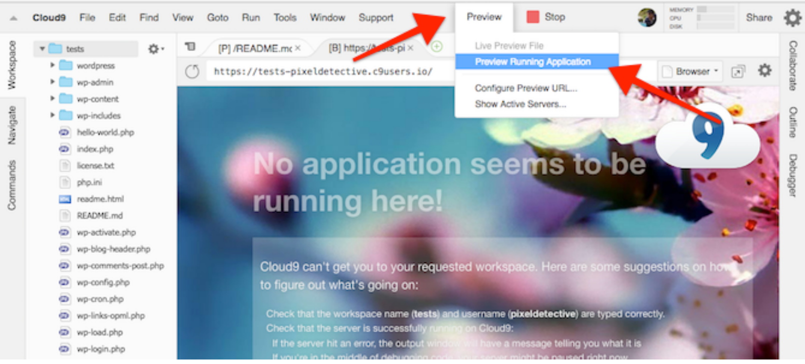
- Check that the workspace name (**tests**) and username (**pixeldetective**) are typed correctly.
- Check that the server is successfully running on Cloud9:
  - If the server hit an error, the output window will have a message telling you what it is
  - If you're in the middle of debugging code, your server might be paused right now
  - The server might be running on a different port; make sure it's on port $PORT with $IP as the IP address

Status Page | Support | Dashboard | Home

Command: mysql-ctl start

```
* Starting MySQL database server mysqld                                [ OK ]
```

Step 7: The server is now running. Click on **Preview** and select **Preview Running Application**.



https://tests-pixeldetective.c9users.io/

No application seems to be running here!

Cloud9 can't get you to your requested workspace. Here are some suggestions on how to figure out what's going on:

Check that the workspace name (**tests**) and username (**pixeldetective**) are typed correctly.
Check that the server is successfully running on Cloud9:
If the server hit an error, the output window will have a message telling you what it is

Step 8: Your WordPress installation is now running inside the IDE. You can see the setup screen here. Let's take this up

Related Articles

INTERNET

5 Best Blog Sites Other Than WordPress and Blogger

WINDOWS , PROGRAMMING

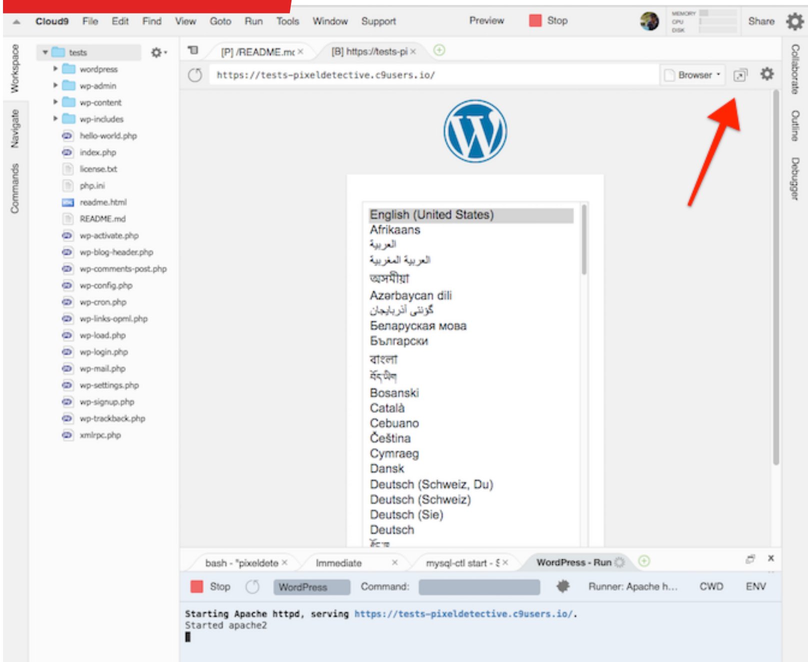
Host and Edit Websites on Windows for Free Using XAMPP

TECHNOLOGY EXPLAINED

7 Reasons to Ditch Your CMS and Consider a Static Site Generator

Latest Deals

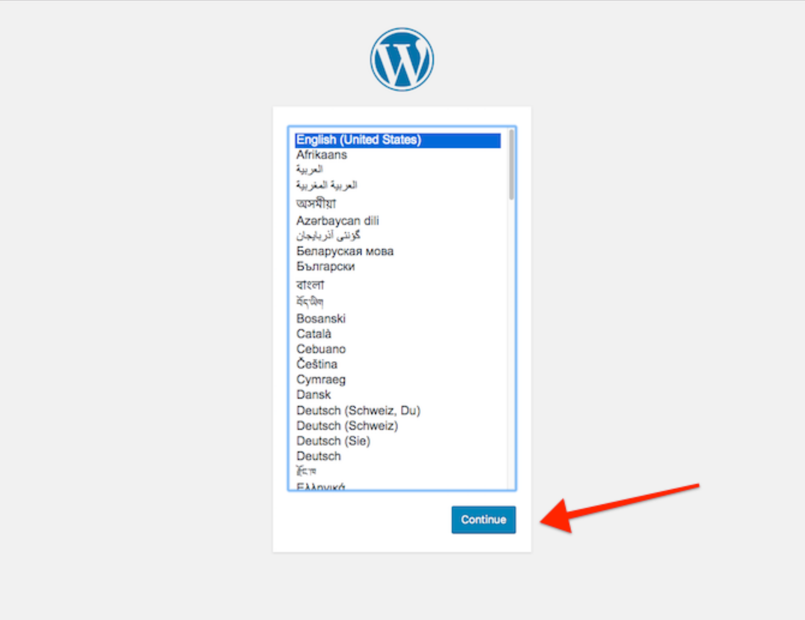
Affiliate Disclosure: By buying the products we recommend, you help keep the lights on at MakeUseOf. .



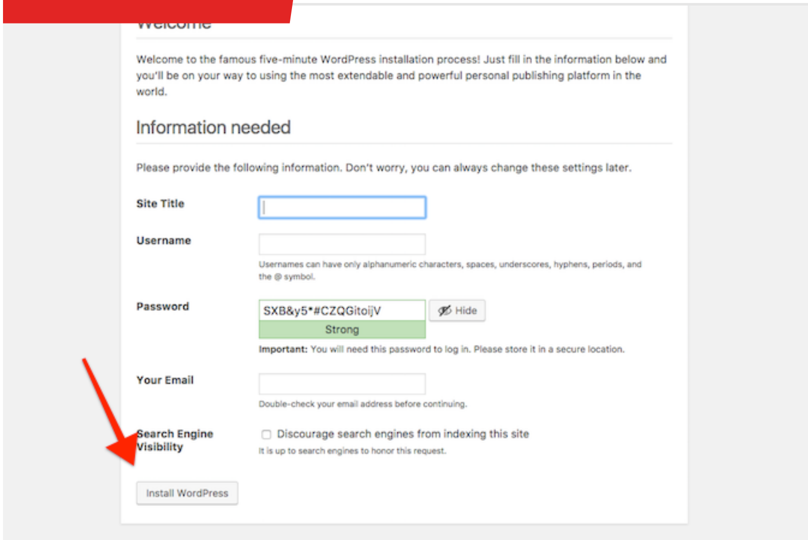
```
Starting Apache httpd, serving https://tests-pixeldetective.c9users.io/.
Started apache2
```

Setting Up Your WordPress Install

Step 1: Now that we're in the full-screen view, select the language of your choice and click on **Continue**.



Step 2: On the Welcome screen, give the site a title, provide the username, password, and your email before clicking **Install WordPress**.



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title

Username

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password SXB&y5*#CZQGitoijV Hide

Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email

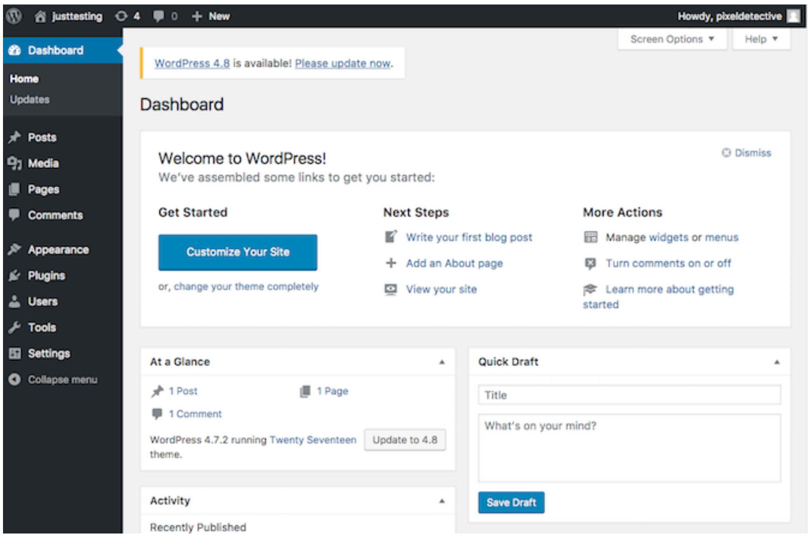
Double-check your email address before continuing.

Search Engine Visibility □ Discourage search engines from indexing this site

It is up to search engines to honor this request.

Install WordPress

Step 3: And there's your WordPress dashboard. You can install themes, customize them, write blog posts, and more.



WordPress 4.8 is available! Please update now.

Dashboard

Welcome to WordPress!

We've assembled some links to get you started:

Dismiss

Get Started

Customize Your Site

or, change your theme completely

Next Steps

Write your first blog post

Add an About page

View your site

More Actions

Manage widgets or menus

Turn comments on or off

Learn more about getting started

At a Glance

1 Post 1 Page

1 Comment

WordPress 4.7.2 running Twenty Seventeen theme. Update to 4.8

Activity

Recently Published

Quick Draft

Title

What's on your mind?

Save Draft

As long as your Cloud9 workspace is live, so is the WordPress install. Yes, this is not a live site, but it's fully equipped for all your experimentation.

Now Level Up Your WordPress Game

A virtual installation is a good playing ground. But as this is WordPress, you'll be able to easily export your customizations from the Cloud9 space and import it into a real, live self-hosted WordPress installation **on a host like Bluehost**.

Image Credits: Maksym Povozniuk/Shutterstock

0 COMMENTS

WRITE A COMMENT

Scroll down for the next article