THE ASYNC INVASION

WHO IS THIS GUY?

MVP Microsoft® Most Valuable Professional

How almost every Stack Overflow question looks to me

I agree ×

How do I do this thing?

43

4

I'm not aware of how to use Google, how do I do this basic thing in Language X?

coding question

share | improve this question

edited Apr 2 '12 at 8:13
Grammar Nazi
2.5M ●7 ●39 ●70

asked Feb 1 '10 at 16:27
1337z0r
2 ●1 ●3 ●6

tagged

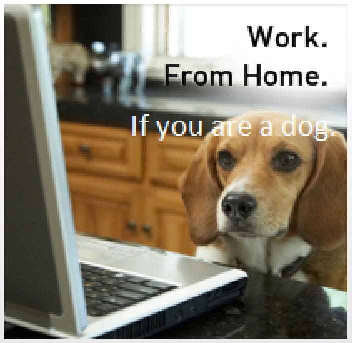
coding × 155474

question × 37256

asked 3 years ago
viewed 43962 times
active 1 month ago

Community Bulletin

event **Microsoft can help you port your app to Windows 8 - win prizes with Apptivate!**
– now through June 7



CAREERS 2.0
by stackoverflow

5 Answers

active oldest votes

12

Lazy but functional answer with no extensibility

share | improve this answer

answered Feb 1 '10 at 16:30
Joe the Coder
1,230 ●3 ●14 ●25

That's perfect! I'm never checking back here again! – 1337z0r Feb 1 '10 at 16:42

248

A lot of people think you should do it in a lazy way, however in the long run it will help you if you read this well eloquent wall of text that acutely describes problems you will inevitably face but not take the time to read about here; enjoy these code samples and illustrations I pulled from thin air anyway!

share | improve this answer

answered Feb 1 '10 at 16:30
WTF look at my points
34M ●400 ●150 ●60

18

The official way to do this is link.

share | improve this answer

answered Feb 1 '10 at 16:29
Professional Coder
5,241 ●2 ●24 ●63

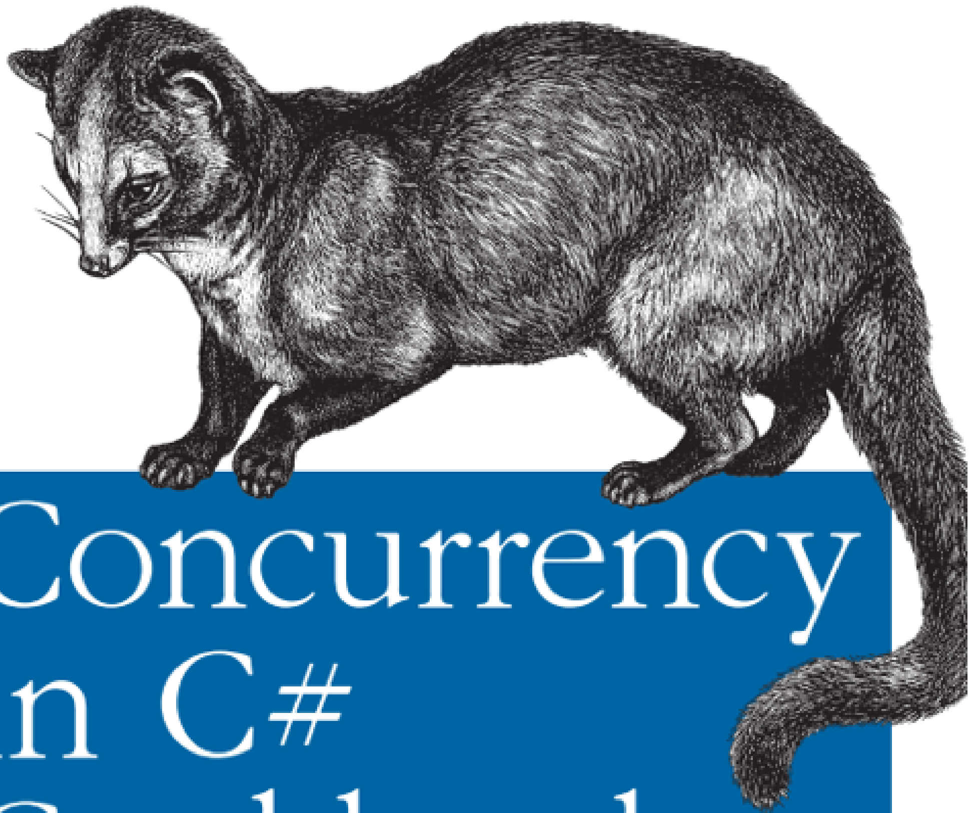
Uh.. No thanks, this is too hard. Can you give me an example of how my code should look when complete? – 1337z0r Feb 1 '10 at 16:41

CAREERS 2.0

There really are jobs for coders available! Not for you though.
Kiersted Systems
Houston, TX / relocation

Highly paid, competitive benefits; that one language you didn't learn.

Asynchronous, Parallel, and Multithreaded Programming



Concurrency in C# Cookbook

O'REILLY®

Stephen Cleary

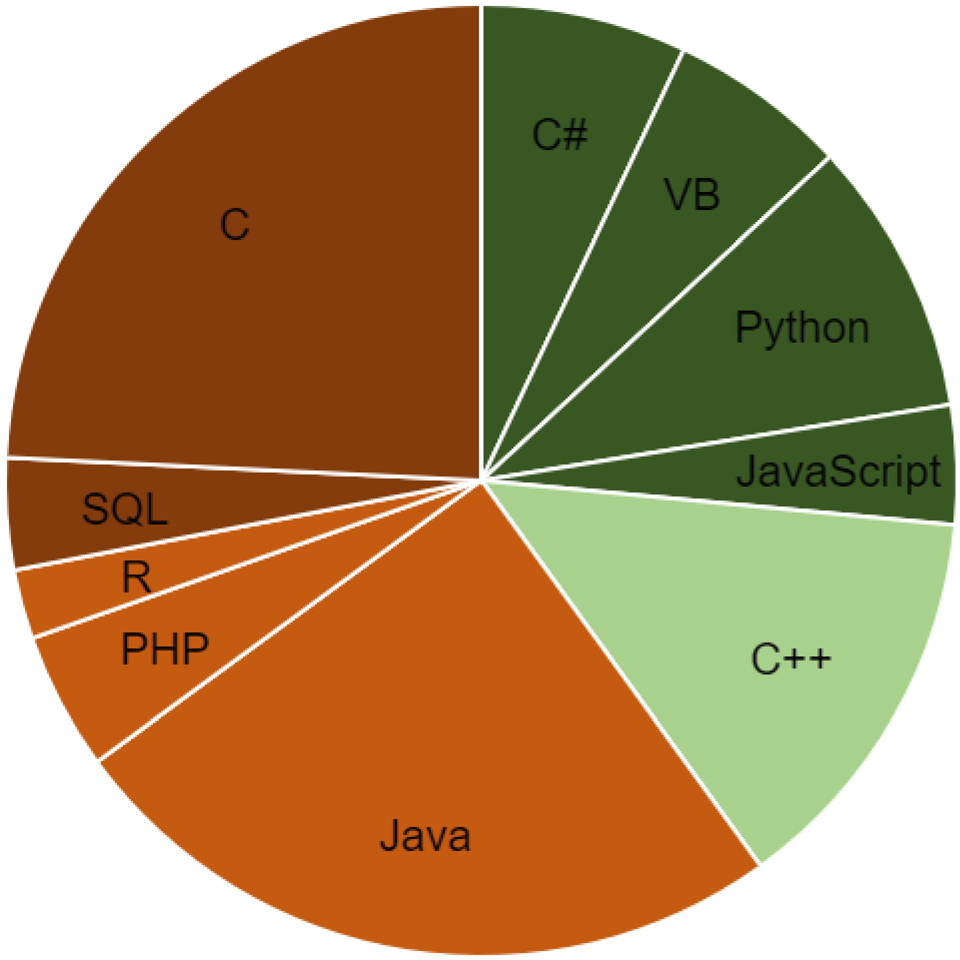
THE ASYNC REVOLUTION!

THE ASYNC ~~REVOLUTION!~~ INVASION?

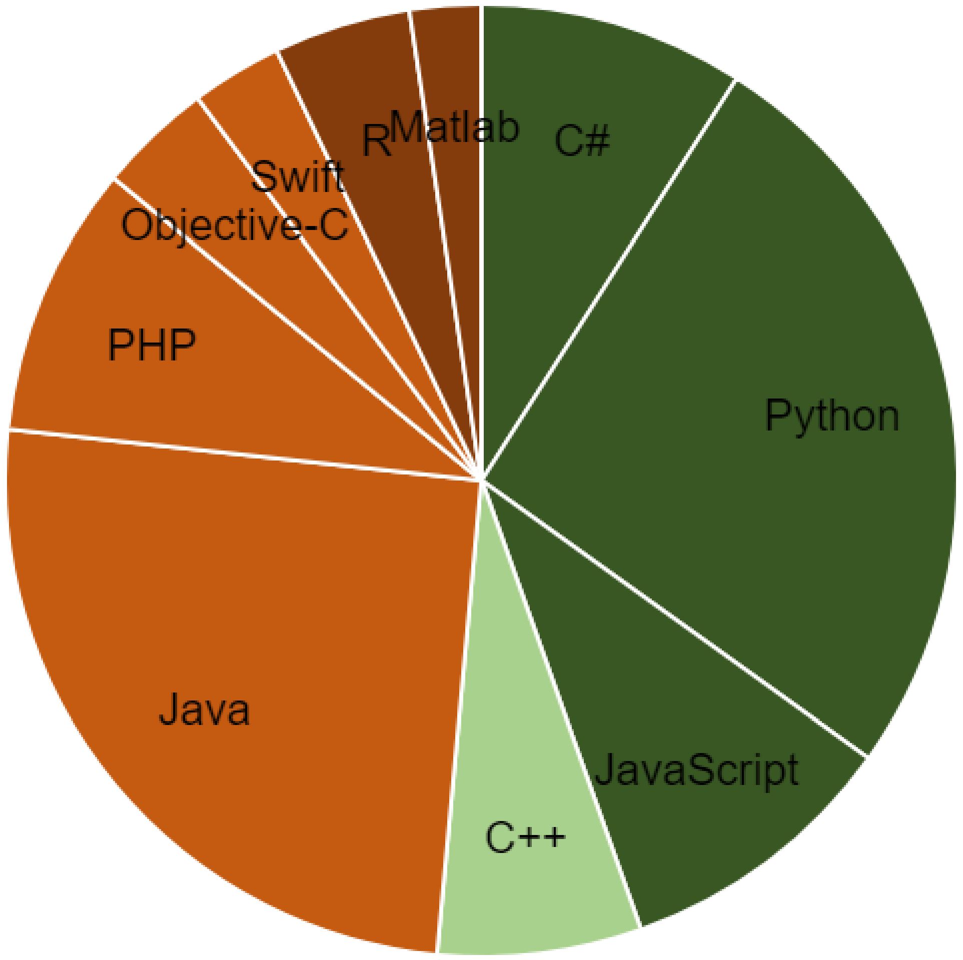
FUTURE TIMELINE?

- C++ (n4680 Coroutines) – C++20?
- Kotlin (experimental coroutines in 1.1)
- Rust (nightly since 2017-08)

TIOBE, 2018-06



PYPL, 2018-06



MUA HA HA HA HA!!
imgflip.com

TERMINOLOGY

What does "asynchronous" really mean?

- Concurrent
  - Multithreaded
    - Parallel
  - Asynchronous

YOU'RE TELLING ME
I CAN BE CONCURRENT
WITHOUT
THREADS?
imgflip.com

There Is No Thread

For I/O, which is more natural?

- Synchronous APIs
- Asynchronous APIs

MIND: BLOWN

WHY?

BENEFITS OF ASYNCHRONY

- UI: responsiveness.
- Server: scalability.

WHY IS IT IMPORTANT TODAY?

- Mobile
- Cloud

BUT WHY ASYNC/AWAIT?

ASYNCHRONY: AN ARCHAEOLOGICAL TOUR

- Events
- Callbacks / CPS
- Futures
- Async/Await

COMPANY CONFIDENTIAL

Our application will...

1. Download a string from teh internets
2. Save it to a database

SYNCHRONOUS SOLUTION

PROVIDED API

```
string Download();
void Save(string);
```

IMPLEMENTATION

```
void DownloadAndSave() {
  string data = Download();
  Save(data);
}
```

EVENTS

PROVIDED API

```
void Download();
event<string> DownloadCompleted;

void Save(string);
event<void> SaveCompleted;
```

SECRET SAUCE API

```
void DownloadAndSave();
event<void> DownloadAndSaveCompleted;
```

EVENTS: IMPLEMENTATION

```
event<void> DownloadAndSaveCompleted;
void DownloadAndSave() {
  DownloadCompleted += downloadResult => {
    if (downloadResult.error) {
      trigger DownloadAndSaveCompleted with downloadResult.error;
      return;
    }
    SaveCompleted += saveResult => {
      if (saveResult.error) {
        trigger DownloadAndSaveCompleted with saveResult.error;
        return;
      }
      trigger DownloadAndSaveCompleted;
    };
    Save(downloadResult.data);
  };
  Download();
}
```

EVENTS: PROBLEMS

- Have to read code backwards.
- Manual error handling.
- Deep nesting.
- Non-trivial logic (loops, joins) require manual state machines.

CALLBACKS / CPS

PROVIDED API

```
void Download(callback<string>);
void Save(string, callback<void>);
```

SECRET SAUCE API

```
void DownloadAndSave(callback<void>);
```

CALLBACKS / CPS: IMPLEMENTATION

```
void DownloadAndSave(callback<void> cb) {
  Download(downloadResult => {
    if (downloadResult.error) {
      cb(error = downloadResult.error);
      return;
    }
    Save(downloadResult.data, saveResult => {
      if (saveResult.error) {
        cb(error = saveResult.error);
        return;
      }
      cb();
    });
  });
}
```

CALLBACKS / CPS: PROBLEMS

- ~~Have to read code backwards.~~
- Manual error handling.
- Deep nesting.
- Non-trivial logic (loops, joins) require manual state machines.

FUTURES

A “Future” represents a future value.

Futures complete exactly once, either with a value or with an error.

Futures support continuations.

Futures are object representations of asynchronous operations.

Futures are monads.

A "Future" can be anything...

- File download
- Database write
- Timeout
- "Join" of other futures
- Mutual exclusion

FUTURES

PROVIDED API

```
Future<string> Download();
Future<void> Save(string);
```

SECRET SAUCE API

```
Future<void> DownloadAndSave();
```

FUTURES: IMPLEMENTATION

```
Future<void> DownloadAndSave() {
  return Download()
      .then(data => { return Save(data); });
}
```

FUTURES: PROBLEMS

- ~~Have to read code backwards.~~
- ~~Manual error handling.~~
- ~~Deep~~ Shallow nesting.
- Non-trivial logic (loops, joins) require ~~manual state machines~~ multiple methods.

ASYNC / AWAIT

PROVIDED API

```
Future<string> Download();
Future<void> Save(string);
```

SECRET SAUCE API

```
Future<void> DownloadAndSave();
```

ASYNC/AWAIT: IMPLEMENTATION

```
Future<void> DownloadAndSave() {
  string data = await Download();
  await Save(data);
}
```

```
void DownloadAndSave() {
  string data = Download();
  Save(data);
}
```

FUTURES: PROBLEMS

- ~~Have to read code backwards.~~
- ~~Manual error handling.~~
- ~~Deep nesting.~~
- ~~Non-trivial logic (loops, joins) require state machines or multiple methods.~~

SUMMARY

Asynchrony is important today...

Because of cloud and mobile...

But asynchronous code is hard...

So languages are adopting `async/await`...

To make asynchronous code easier.

Q&A

Image from Etsy user Rosewine
used with permission