T15

Test Automation

2019-05-02 13:30

API Testing: Going from Manual to Automated

Presented by:

Patrick Poulin

API Fortress

Brought to you by:

888-268-8770 · 904-278-0524 - info@techwell.com - http://www.stareast.techwell.com/

Patrick Poulin

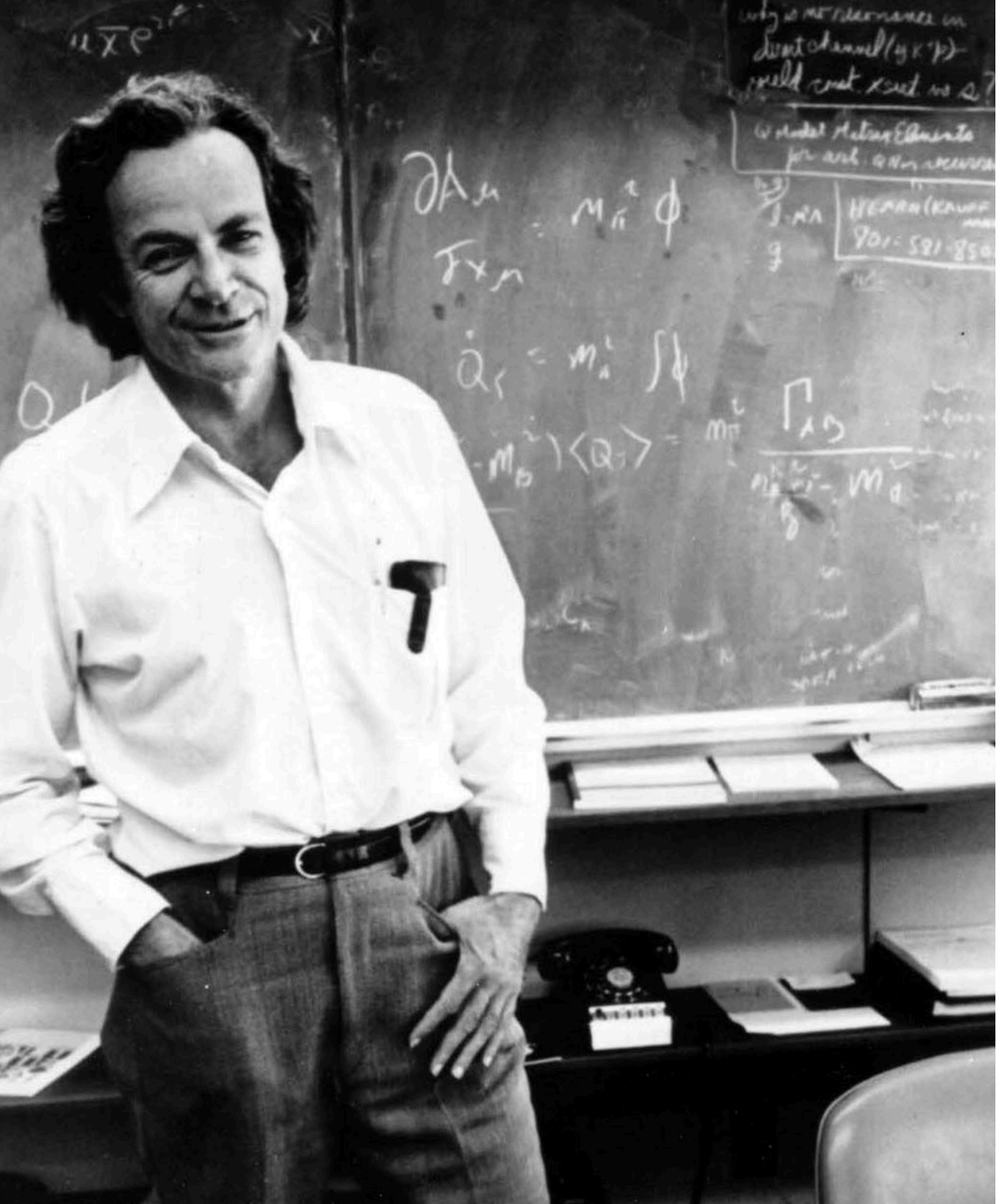
Patrick Poulin has been working in NYC tech for 12 years. His career started as a project manager, but soon he was managing the retail vertical for a company building the first mobile websites for over 75 major brands such as Tesco, Target, Macys, and MAC Cosmetics. After a (thankfully) short stint in adtech, he became the API Evangelist for Getty Images. This is where he first recognized the lack of good API tools. That experience is what led to the creation of API Fortress with his cofounder.

Watching the World Burn:

API Testing as an Afterthought

With APIFORTRESS

Jason Ioannides, Sales Engineer



Why Are We Asking the Questions We Ask With Testing?

Richard Feynman
1918-1988
Theoretical Physicist

How Does Feynman Apply?

“Does the software work?”

How are we asking this question and more importantly, why?

Unit Testing

Our example (a real snippet!):

```
def calculate_final_price(price, discount):
    if discount=='30%':
        return price-(price*0.3)
    if discount=='50%':
        return price/2
    return price
```

Manual Front End Testing

Visual Regression Testing

“Click” Testing

API Testing

Request: POST /cart/add

```
{
 "productId": 511,
 "name": "Smarty pants",
 "quantity": 1,
 "price": 5.99,
 "discount": "30%"
}
```

Response:

```
{
 "transaction": "ok",
 "subtotal": 4.193
}
```

Request: POST /cart/add

```
{
 "productId": 511,
 "name": "Smarty pants",
 "quantity": 1,
 "price": 1.99,
 "discount": "50%"
}
```

Response:

```
{
 "transaction": "ok",
 "subtotal": 0.995
}
```

Cardinal Rule #1

We are not simply testing communication.

APIs are complex beasts and can expose
a huge volume of internal functionality.

Cardinal Rule #2

We have to treat API testing as the special sort of situation that it is.

Testing internal functionality of software while the whole system is operating.

“Taking the pulse” of the software.

Rule 1: **Keep it DRY**

DRY:

Don’t Repeat Yourself

- Parameterize data and code wherever you can
- Increase the modularity of the work you’re doing
- Create reusable tests

Rule 2: **Make Your Intentions Clear**

- How am I going to remember what this test is for?
- How is the next developer or tester going to know what this test is for?
- We need to do increasing amounts of work if our tests can’t be reused
- Collaboration is very difficult when the intent behind a test is obscured

Rule 3: **Act Like the Consumer Would!**

- Stop creating test cases in a vacuum
- Many APIs exist in the scope of a system
- Testing individual parts is important, but testing full workflows is more important
- Simulate user workflows to find the holes

Rule 4: **Eliminate Static Data Sources!**

- Static data is rarely your friend
- Caveat: Some endpoints rely on static data. That’s okay!
- Use live data wherever possible
- Where live data is unavailable, use mock responses that match your expected response

+

APIFORTRESS

LEARN MORE AT:

www.apifortress.com/axway