

how to make APIs that people like

who uses your APIs?

lazy people, that's who

nobody reads documentation

nobody reads documentation

intuitive

developers are users
APIs are UX

guessabilize and explorabilize and documentize

wtf?

Every time a user of your APIs guesses wrongly about how to use them, it is your fault.

browseable

```
{
  "total": 3,
  "results": [
    {
      "album": "Dummy (non uk version)",
      "provider_album_id": "7digital:album:12142:GB",
      "purchase_url": "u1ms://musicsearch.ubuntu.com/v1/tracker?url=http%3A%2F%2Fstores.7digital.com%2F7_1%2Fartists%2Fportishead%2Fdummy-non-uk-version%3Fpartner%3D983",
      "artist": "Portishead",
      "image": "http://cdn.7static.com/static/img/sleeveart/00/000/121/0000012142\_50.jpg",
      "title": "Roads",
      "provider_track_id": "7digital:track:123521:GB",
      "source": "Ubuntu One Music Store",
      "details": "http://musicsearch.ubuntu.com/v1/track?geo\_store=GB&id=7digital%3Atrack%3A123521%3AGB",
      "year": "2005",
      "type": "track",
      "web_purchase_url": ""
    },
  ]
}
```

consistency

REST.. ish



<http://mything.example.com/API?action=add>

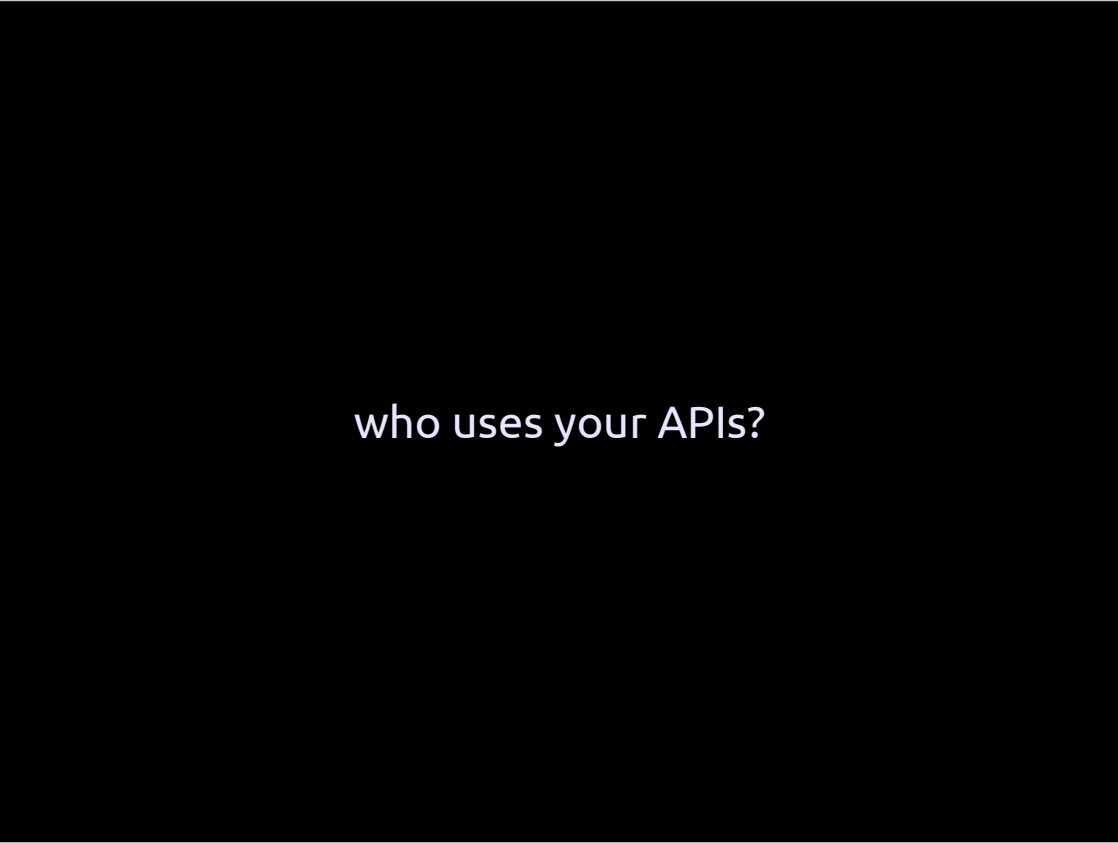
explorability
guessability
browseability
consistency

explorability
guessability
browseability
consistency
documentation

@sil

how to make APIs that people like

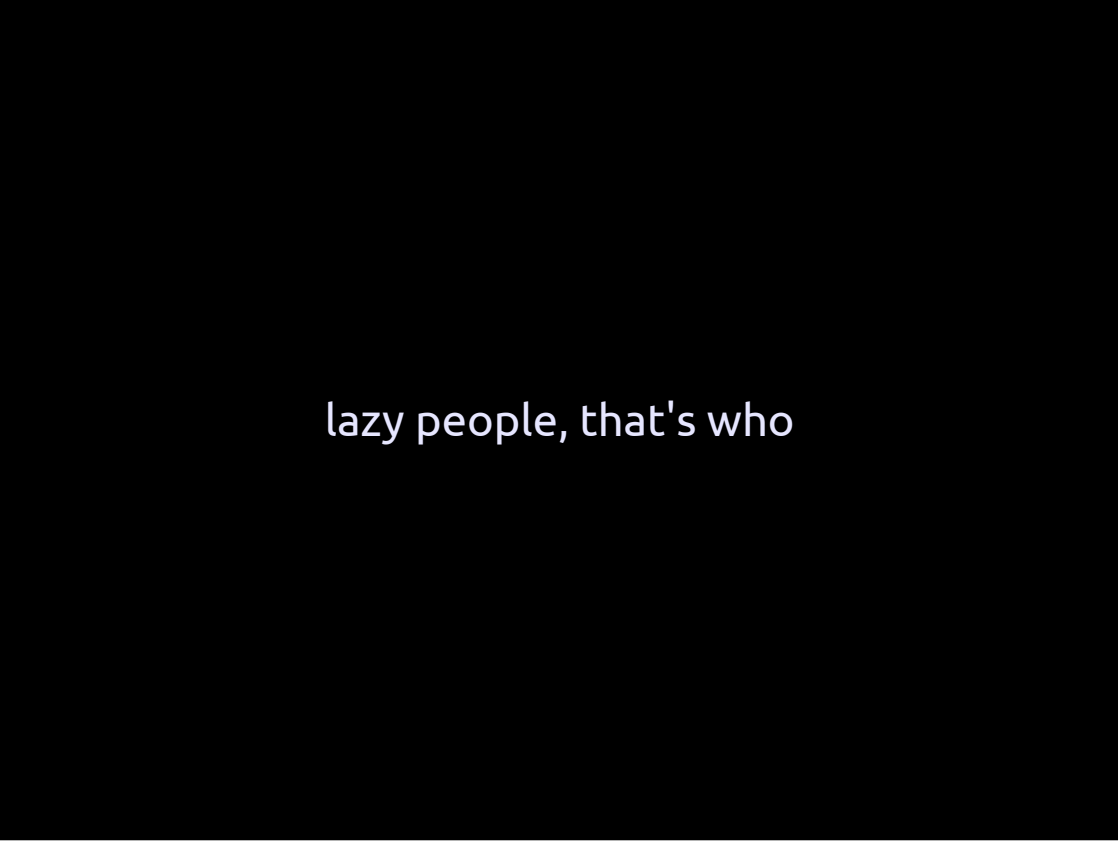
Stuart Langridge, Multipack Show and Tell, May 2012



who uses your APIs?

I'm going to talk about HTTP APIs here, but most of it applies to most stuff.

Who here uses HTTP APIs for something, either as a provider or a consumer?



lazy people, that's who

That's you, users. Lazy.

Laziness is not a bad thing.



nobody reads documentation

The most important point, and you all already know this point, is this: nobody reads documentation.

You are thinking: that's not the case, not always.

nobody reads documentation

NOBODY reads it.




intuitive

The most important thing we can do with APIs is to make them intuitive.

Remember: developers are users too.

Your APIs have a user experience, just like your UIs do. Everything about UX design applies here as well.



developers are users
APIs are UX

Developers are users, too. They're users of your APIs. Everything you know about UX design applies to your APIs too.

Make them guessable and explorable. Guessabilize and explorabilize and documentize them.

guessabilize and explorabilize and documentize

wtf?

None of those are words. But you knew what I meant. People are very good at extending their knowledge into new areas by generalising, by intuiting that if it does *this* and *this* then it'll do *this* as well.

This is why nobody reads documentation. Everyone reads **some** of the documentation and then generalises the knowledge.

Every time they guess wrong, it is your fault.

Every time a user of your APIs guesses wrongly about how to use them, it is your fault.

Seems harsh, I know.

Unlucky. No-one said this was going to be easy.



browseable

We build things for the browser. APIs are like that too.

Make them work in the browser. That means you can get to them with cookie authentication or at a pinch basic auth over https; not just a token.

Curl is a shitty UX.

Make them cross-linked. Explorable.

```
{
  "total": 3,
  "results": [
    {
      "album": "Dummy (non uk version)",
      "provider_album_id": "7digital:album:12142:GB",
      "purchase_url": "u1ms://musicsearch.ubuntu.com/v1/tracker?url=http%3A%2F%2Fstores.7digital.com%2F7_1%2Fartists%2Fportishead%2Fdummy-non-uk-version%3Fpartner%3D983",
      "artist": "Portishead",
      "image": "http://cdn.7static.com/static/img/sleeveart/00/000/121/0000012142\_50.jpg",
      "title": "Roads",
      "provider_track_id": "7digital:track:123521:GB",
      "source": "Ubuntu One Music Store",
      "details": "http://musicsearch.ubuntu.com/v1/track?geo\_store=GB&id=7digital%3Atrack%3A123521%3AGB",
      "year": "2005",
      "type": "track",
      "web_purchase_url": ""
    }
  ]
}
```

So links are clickable. JSON is readable.

consistency

Consistency is there for guessability. It's not about being perfect; it's about helping other people to get into your mindset. You should think like them, but unless you're the best psychologist ever, meet them halfway; try and get them to think like you.



REST... ish

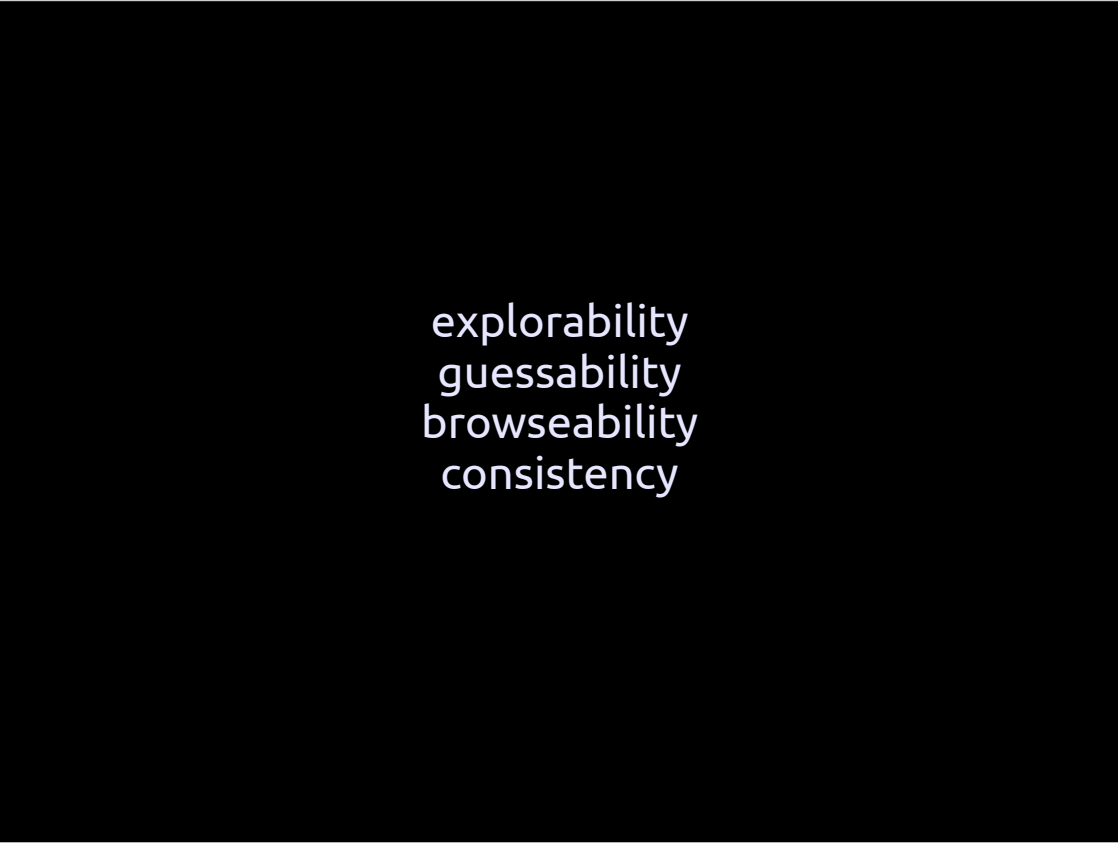
This is why RESTish APIs are a good idea. You GET from one place and PUT back to the same place.

Don't go overboard, but don't do this:



Don't.

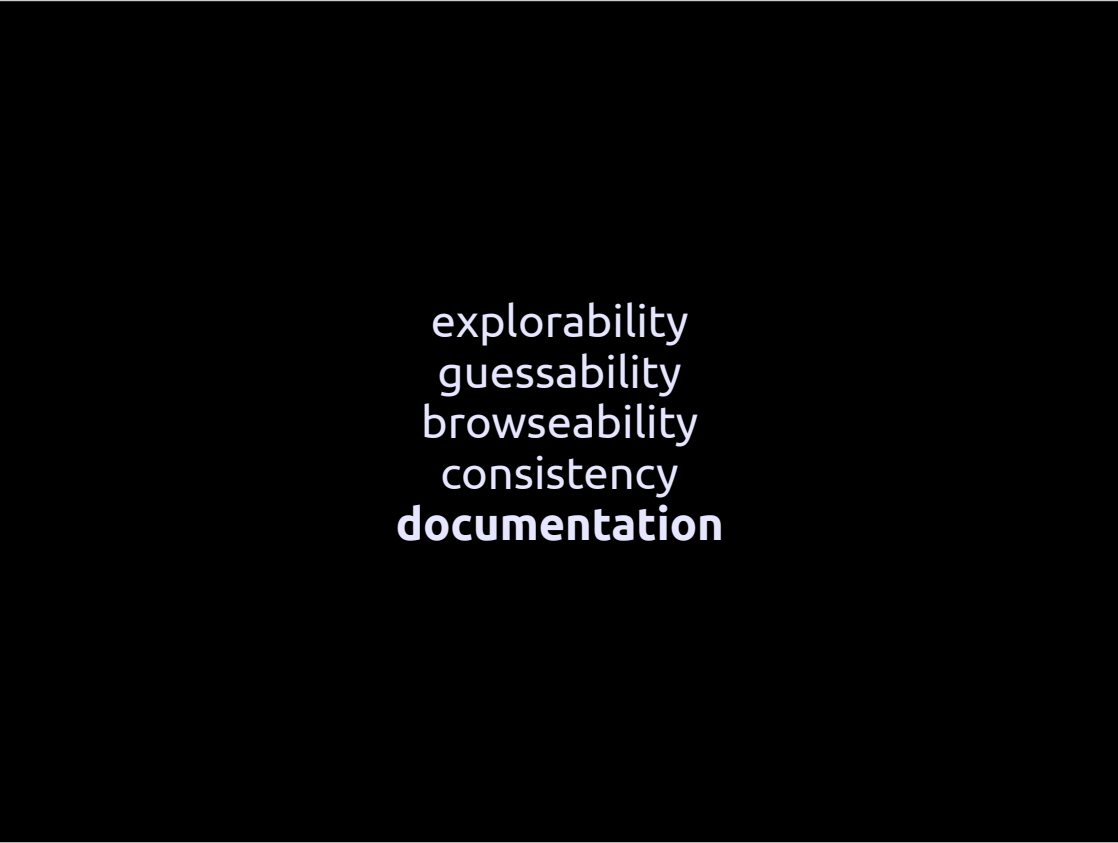
It is impossible to guess what all the actions are.



explorability
guessability
browseability
consistency

With U1DB we've argued tons about things like whether `get_values_from_index` or `get_index_values` is the most appropriate name. We've done user research. This stuff is important.

So we've talked about these... what else?



explorability
guessability
browseability
consistency
documentation

I know nobody reads it.

But if your APIs are already guessable and explorable and browseable and consistent, then your documentation does two things: it gives people a reference point to cling to, and it's advertising for your services.

If you've got good examples, your docs will come up a lot in searches!



@sil

And when you're done, tell me about it.

Thank you.