Websockets and chatbots

PyBCN November 2017 – Carlos Fenollosa

Who am I?

- Computer Engineer, MSc in Artificial Intelligence
- Barcelona Supercomputing Center – Biotech researcher
- Pythonista since 2011

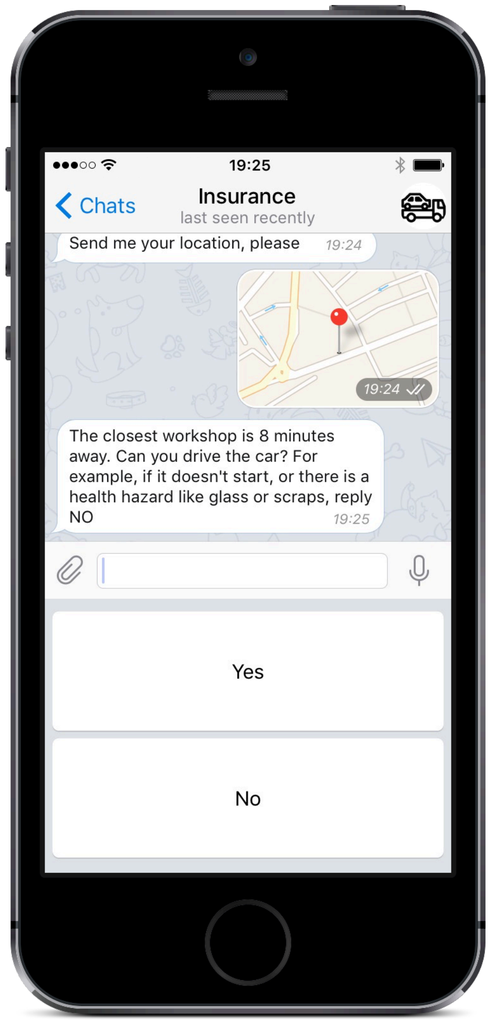
- Founder Puput – cool tech but bad market fit, it failed
- *Chatbots*
- Founder Optimus Price – Automatic pricing powered by AI
    - Reinforced learning, true AI
    - Full Python stack
    - Small team, great engineers, normal people (SiliconValleyHBO is not a parody)

Chatbots

- Simple definition: they reply to the user input
- Wide range of “intelligence”
  - Dumb bots that follow scripts (if-else)
  - Basic understanding of concepts
  - Conversation context / proactive
- Uncanny valley – manage expectations!
- Not necessarily a personal assistant, many use cases
- `M-x doctor`

Why chatbots?

- **<u>The best interface is no interface</u>**
- No need to install an extra app
- People already know how to chat
- Much better than the traditional "wizard"
- Chat != text. Bots allow for multimedia, voice, buttons…



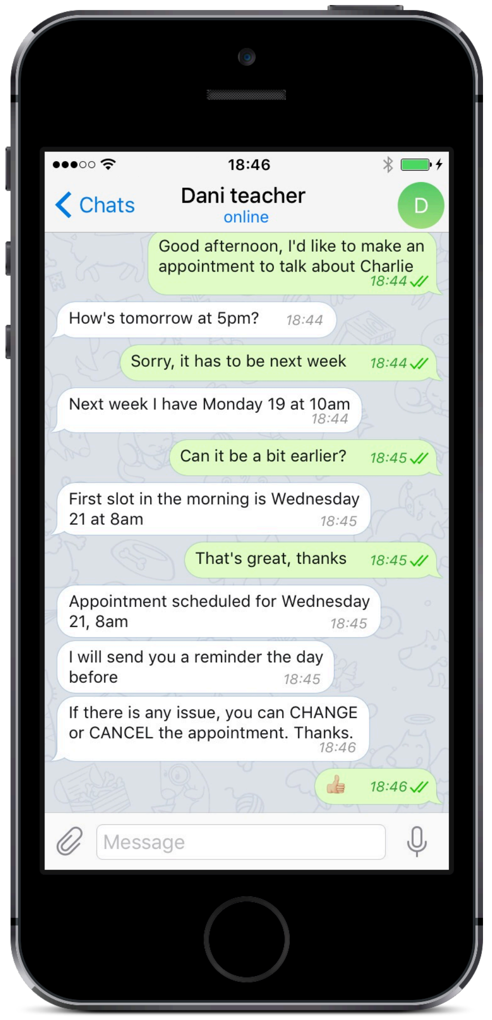
19:25

Chats — Insurance — last seen recently

Send me your location, please 19:24

19:24

The closest workshop is 8 minutes away. Can you drive the car? For example, if it doesn't start, or there is a health hazard like glass or scraps, reply NO 19:25



18:46

Chats — Dani teacher — online

Good afternoon, I'd like to make an appointment to talk about Charlie 18:44

How's tomorrow at 5pm? 18:44

Sorry, it has to be next week 18:44

Next week I have Monday 19 at 10am 18:44

Can it be a bit earlier? 18:45

First slot in the morning is Wednesday 21 at 8am 18:45

That's great, thanks 18:45

Appointment scheduled for Wednesday 21, 8am 18:45

I will send you a reminder the day before 18:45

If there is any issue, you can CHANGE or CANCEL the appointment. Thanks. 18:46

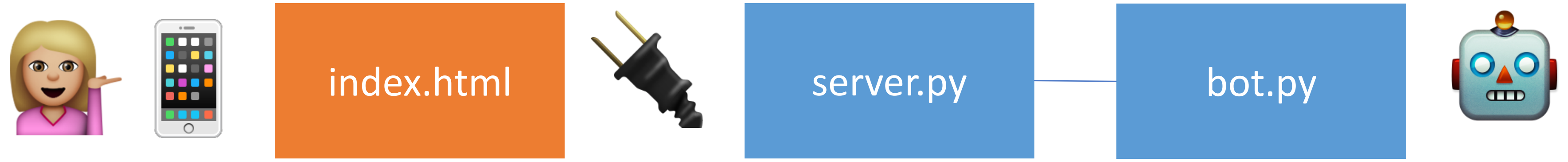
👍 18:46

Client-Server HTTP communication

- 1990s: HTML Form + CGI
  - Synchronous for client (submit + page reload)
  - Push
- 2005+: AJAX + CGI
  - Asynchronous for client thanks to JS
  - Push
  - Long polling: channel open until timeout (60s) + re-open it
- 2012+: Websocket (browser) + Websocket (server)
  - Persistent
  - Bidirectional

Today's project

1. Simple bot (`bot.py`)
2. Create a websocket server (`server.py`)
3. Create a websocket client (`index.html`)
4. Advanced bot techniques



bot.py

```python
def reply(msg):
    return 'You said "{}"'.format(msg)
```

server.py

```python
import asyncio
import websockets



start_server = websockets.serve(handler, '0.0.0.0', 1234)
asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

server.py

```python
import bot


@asyncio.coroutine
def handler(websocket, path):
    while True:
        try:
            msg = yield from websocket.recv()
            reply = bot.reply(msg)
            yield from websocket.send(reply)
        except websockets.ConnectionClosed:
            return
```

```python
import asyncio
import websockets
import bot


@asyncio.coroutine
def handler(websocket, path):
    while True:
        try:
            msg_str = yield from websocket.recv()
            reply = bot.reply(msg_str)
            yield from websocket.send(reply)
        except websockets.ConnectionClosed:
            return

start_server = websockets.serve(handler, '0.0.0.0', 1234)
asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

index.html

```
var ws = new WebSocket('ws://127.0.0.1:1234/');

ws.onmessage = function (event) {
    var msg = event.data;
    console.log('Recv: ' + msg);
};
```

index.html

```html
<input type='text' id='msg_input'>
<input onclick='send()' type='button' value='Send'>
```

```javascript
function send() {
      var msg = document.getElementById('msg_input').value;
      console.log('Sent: ' + msg);
      ws.send(msg);
}
```

```html
<!DOCTYPE html>
<html>
    <body>
        <input type='text' id='msg_input'>
        <input onclick='send()' type='button' value='Send'>

        <script type="text/javascript">
            var ws = new WebSocket('ws://127.0.0.1:1234/');

            ws.onmessage = function (event) {
                var msg = event.data;
                console.log('Recv: ' + msg);
            };

            function send() {
                var msg = document.getElementById('msg_input').value;
                console.log('Sent: ' + msg);
                ws.send(msg);
            }
        </script>
    </body>
</html>
```

Advanced bot techniques

- Word declination (Stemming)
  - Necessary to find words in texts regardless of declination
- Create credible nonsense (Markov chains)
  - Fun
- NLP: Google Cloud, MS Azure, IBM Bluemix
  - Semantics!
  - Entity recognition
  - Sentiment
  - Not for this talk

Stemming

```
>>> from nltk.stem.snowball import SpanishStemmer
>>> ss = SpanishStemmer()
>>> ss.stem('amigos')
'amig'
>>> ss.stem('casas')
'cas’
>>> ss.stem('ordenadores')
'orden'
```

```
>>> from nltk.stem.snowball import EnglishStemmer
>>> es = EnglishStemmer()
>>> es.stem('woman')
'woman'
>>> es.stem('houses')
'hous'
>>> es.stem('women')
'women'
>>> es.stem('cards')
'card'
```

Tips for good stemming

1. Tokenize

```
>>> nltk.word_tokenize("I like long sentences. Let's write software.", "english")
['I', 'like', 'long', 'sentences', '.', 'Let', "'s", 'write', 'software', '.']
```

2. Remove accents (optional, but sometimes useful)

```
>>> import unicodedata
>>> nfkd = unicodedata.normalize('NFKD', 'áéíóúçñ')
>>> [c for c in nfkd if not unicodedata.combining(c)]
['a', 'e', 'i', 'o', 'u', 'c', 'n']
```

Tips for good stemming

3. Lowercase

```
>>> 'I HATE CAPS LOCK'.lower()
'i hate caps lock'
```

4. Put it all together

```
>>> tokens = nltk.word_tokenize('Los adolescentes van con malas compañías'.lower(),
'spanish')
>>> stems = [ss.stem(token) for token in tokens]
>>> ss.stem('adolescente') in stems
True
>>> ss.stem('compañía') in stems
True
```

Markov chains

- Chain words that are usually together, e.g.
  - `football → [match, field, player]`
  - `art → [gallery, competition, exhibition]`
  - Polisemics! `football → match → cigarrette`

- SCIgen - An Automatic CS Paper Generator
https://pdos.csail.mit.edu/archive/scigen/

- RajoyPresidente.org (now defunct)

Markov chains

1. Create chain

```
>>> from pymarkovchain import MarkovChain
>>> mc_corpus = MarkovChain('/tmp/mc_corpus')
```

2. Generate with a corpus

```
>>> import nltk
>>> mc_corpus.generateDatabase(' '.join(word.lower().replace('_', ' ')
             for word in nltk.corpus.cess_esp.words() if word[0].isalpha()))
>>> mc_corpus.dumpdb()
```

3. Have fun!

```
>>> mc_corpus.generateStringWithSeed('edificio')
'edificio democrático contra este partido también se muestra generoso para
ganar en michigan el gobernador mostró a los millones en lo mismo a
florentino ariza supiera de quién hablaba temiendo una revelación'
```

Bot tips

- Bots are much more fun with friends
  - Markov chains with corpus extracted from chat history
  - Connect your bot with public APIs: image/speech recognition, telephony, memes
  - Add randomness!
  - Give it a personality
  - Find creative uses (DJ Roberto Bonio)

- How to distribute your bot
  - Web page (websockets)
  - Telegram, Facebook, Twitter (use a client library)

Parting thoughts

- **Use python 3 for new projects!**
  - Worth it just for Async, Unicode
  - Legacy code? `$ sed 's/print \(.*\)$/print(\1)/’`

- Pay for good tools, it’s worth it

- Use jupyter notebooks instead of the REPL
  - Good tools integrate notebooks seamlessly ;-)

Thanks for listening!

https://github.com/cfenollosa/chatbot_pybcn

carlos.fenollosa@gmail.com

cfenollosa.com

@cfenollosa