ripple motion
services

Créateur d’applications mobiles

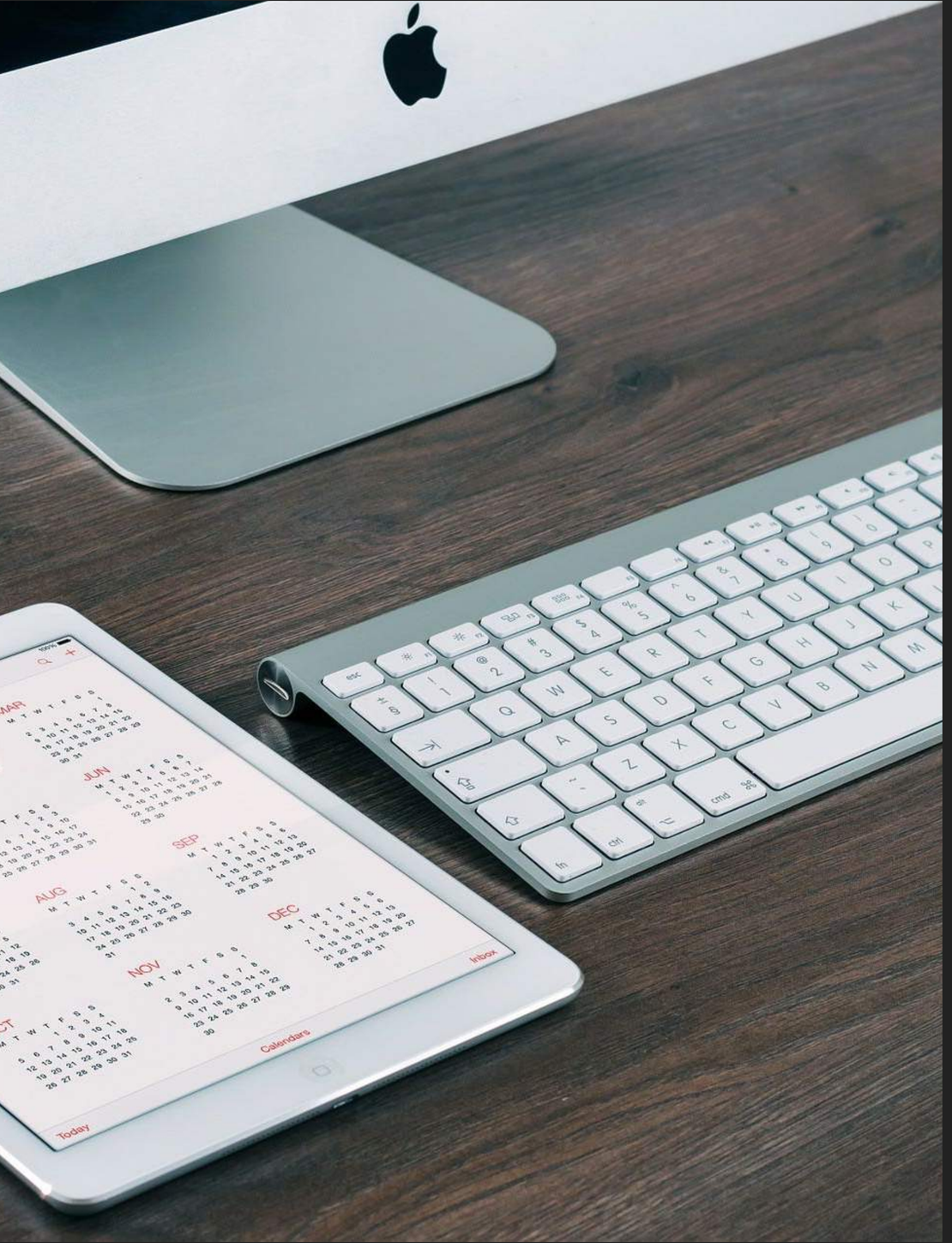
TECH TALKS

GRATUITS & OUVERTS AU PUBLIC

Tech talks sous forme de courtes formations
le **mercredi** de **17h** à **18h** !

N’hésitez pas à nous **contacter** afin de participer
aux **prochaines sessions** au sein de nos locaux !

ripple motion services



TECH TALK #14 – 01.03.2017

CELERY BASICS FOR DJANGO

ripple motion services

ALIX MARTINEAU
DÉVELOPPEUR

WHAT IS CELERY?

SINCE 2009

ASYNCHRONOUS

~~DJANGO ONLY~~

TASK QUEUE BASED ON

DISTRIBUTED

MESSAGE PASSING

ALIX MARTINEAU
DÉVELOPPEUR

WHAT IS CELERY?

ASYNCHRONOUS

TASK QUEUE BASED ON

DISTRIBUTED

MESSAGE PASSING

ALIX MARTINEAU
DÉVELOPPEUR

WHAT IS CELERY?

ASYNCHRONOUS
TASK QUEUE BASED ON
DISTRIBUTED
MESSAGE PASSING

ALIX MARTINEAU
DÉVELOPPEUR

TASK QUEUE

- Used to distribute work across threads, processes or servers, in a non-blocking fashion
- Input is a unit of work called a task
- Worker processes monitor queues for work to do
- Celery supports scheduled and recurring tasks

ALIX MARTINEAU
DÉVELOPPEUR

USE CASES

- Notifying users (email, APNs/GCM notifications…)
- Generating reports (PDFs...)
- Communicating with external services
- Triggering cleanup operations
- Making sure something is done (via *retries*)
- And much more…

ALIX MARTINEAU
DÉVELOPPEUR

WHAT IS CELERY?

ASYNCHRONOUS
TASK QUEUE BASED ON
DISTRIBUTED
MESSAGE PASSING

ALIX MARTINEAU
DÉVELOPPEUR

WHAT IS CELERY?

ASYNCHRONOUS
TASK QUEUE BASED ON
DISTRIBUTED
MESSAGE PASSING

(DISTRIBUTED) MESSAGE PASSING

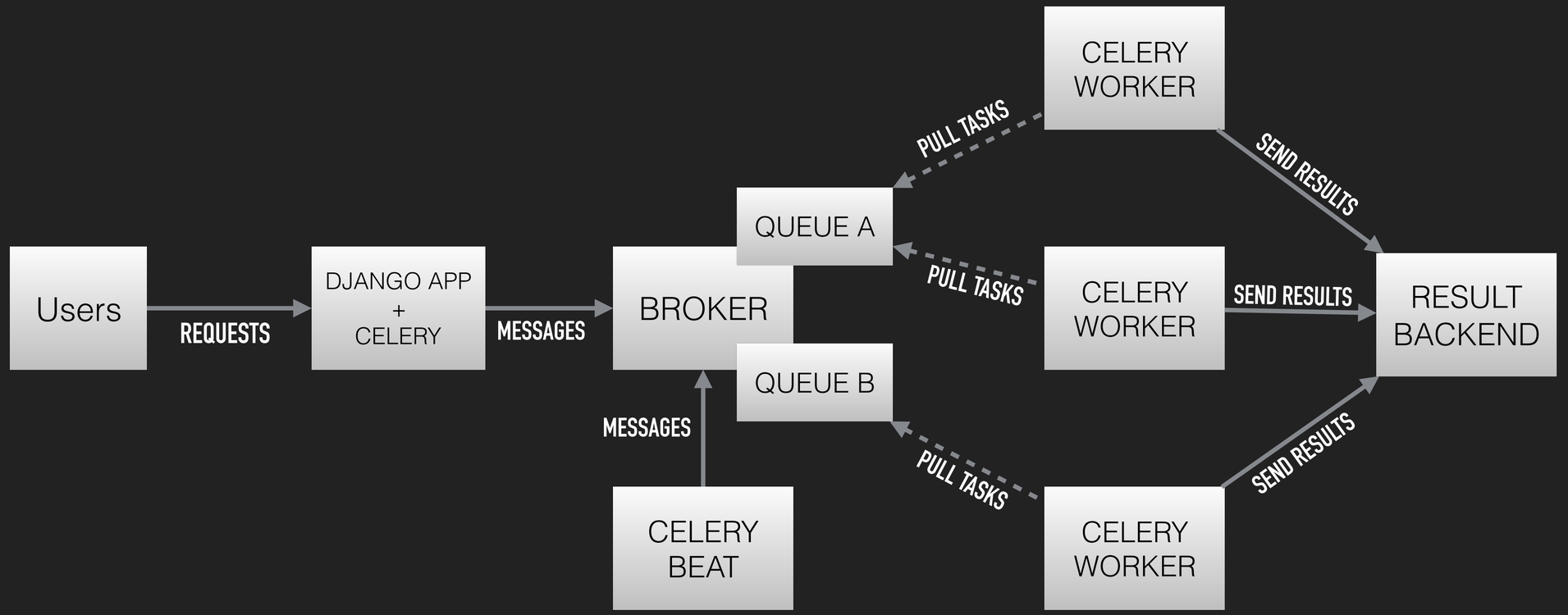
ALIX MARTINEAU
DÉVELOPPEUR

- Celery communicates via messages
- Messages transit through a *broker*
- System can consist of N workers and brokers
  ➡ Scaling and high availability

ALIX MARTINEAU
DÉVELOPPEUR

A TYPICAL SYSTEM

HOW IT ALL FITS TOGETHER



WHAT IS A BROKER?

ALIX MARTINEAU
DÉVELOPPEUR

Broker *verb* — Pr. /ˈbrəʊkə(r)/

broker something to arrange the details of an agreement[…].

— *Oxford Advanced Learner's Dictionary*

- Intermediary role between the different parts of the system
- Answers the question “What remains to be done?”
- Stores queues and associated tasks

ripple motion services

BROKER SUPPORT

ALIX MARTINEAU
DÉVELOPPEUR

- Celery supports lots of different brokers
- Only two are feature-complete and worth your time
  - RabbitMQ
  - Redis

ALIX MARTINEAU
DÉVELOPPEUR

RABBITMQ

RabbitMQ™

- Highly customisable routing
- Persistent queues
- HA features
- Requires a bit of messaging knowledge to use advanced features

ALIX MARTINEAU
DÉVELOPPEUR

REDIS

- You likely already use it
  - ➡ If not: `sudo (apt-get|port) install redis`
- Config is very easy (there is none)
- Can be susceptible to data loss
- Has some quirks(1)

(1) http://docs.celeryproject.org/en/stable/getting-started/brokers/redis.html#caveats

ALIX MARTINEAU
DÉVELOPPEUR

WHICH IS RIGHT FOR YOU?

Simply need to send the occasional email?

➡ Choose redis

Need persistent queues or advanced task distribution?

➡ Choose rabbit face

RESULT BACKENDS

ALIX MARTINEAU
DÉVELOPPEUR

- If you want to keep track of tasks' state or need the return values
- Use cache (redis is good for that)
- Sometimes, you won't need to keep the results
  - ✓ Globally: set `task_ignore_result` to `True` in settings
  - ✓ Per task: use `@app.task(ignore_result=True)`

More about result backends: http://docs.celeryproject.org/en/latest/userguide/tasks.html#result-backends

ALIX MARTINEAU
DÉVELOPPEUR

CONCURRENCY

- Multiprocessing (prefork)
  - ✓ Safe default
  - ➡ Limited to a few proc
- Green threads (E
  - ➡ Bette me cases
  - ⚠ don't perform blocking calls

You c ix both types of workers and route tasks accordi compatibility and/or best performance.

EXPERIMENT

MODERN CELERY

ALIX MARTINEAU
DÉVELOPPEUR

- You don't need *django-celery* anymore
  - ➡ New-style Django integration is very well documented
- Settings have been revamped(1) in Celery 4.0 (current stable)
  - ➡ All Celery-related settings should be prefixed with `CELERY_`
  - ➡ Upgrade with `celery upgrade settings proj/settings.py --django`
  - ➡ Use `app.config_from_object('django.conf:settings', namespace='CELERY')`

(1) full changes: http://docs.celeryproject.org/en/latest/userguide/configuration.html#conf-old-settings-map

ALIX MARTINEAU
DÉVELOPPEUR

TESTING

- Avoid *eager mode* (`CELERY_ALWAYS_EAGER`)
  - ➡ you are only testing an emulation of a worker
- Prefer mocks(1) to test your *tasks* (retries etc.)
  - ➡ `@patch` + `assert_called_with` go a long way
- Test your logic separately

(1) See http://docs.celeryproject.org/en/latest/userguide/testing.html#tasks-and-unit-tests for a mocking example

DOS AND DONTS

ALIX MARTINEAU
DÉVELOPPEUR

- DON'T assume you know *when* tasks will run
  - ✓ Make your scheduled tasks idempotent or use locking
- DON'T pass complex objects to tasks
  - ✓ DO prefer IDs to avoid race conditions
- DO make many small tasks with one purpose each
  - ✓ Single responsibility principle
  - ⚠️ Big tasks are harder to read, maintain and parallelise
- DO *chain* tasks (DON'T make tasks wait for others' results)

ALIX MARTINEAU
DÉVELOPPEUR

KEEP IN MIND

- Use `@shared_task` in reusable apps
- Unlike `runserver`, Celery isn't automatically reloaded on code changes
  - ✓ Don't forget to restart your worker/beat processes
- It's probably a good idea to use exponential back-off for retries

```
def backoff(retries):
    return 2.0 ** retries


@app.task(max_retries=5)
def sync_miserable_pay_account(account_id):
    mp_account = Account.get(id=account_id)
    try:
        mp_account.sync()
    except MiserablePayException as e:
        # Do some cleanup here
        raise self.retry(e, countdown=backoff(self.request.retries))
```

ALIX MARTINEAU
DÉVELOPPEUR

SOME TIPS

- `pip install setproctitle`
  - ✦ Nicer `top` and `ps` output for Celery processes
- `pip install librabbitmq`
  - ➡ If using RabbitMQ (optimised C client)
- `--max-tasks-per-child` to mitigate
  - memory leaks you have no control over
  - your own *bousettes®* 💩

TIPS (CONTINUED)

ALIX MARTINEAU
DÉVELOPPEUR

- Take advantage of Celery's `purge` and `inspect` commands

```
$ celery -A proj purge
$ celery -A proj inspect scheduled
```

- Enable RabbitMQ's *management* plugin for a nice web UI
  - ➡ Useful to monitor RabbitMQ
  - ➡ Makes it easy to manage users, messages, queues etc.

- Flower is a web-based Celery monitoring solution

ALIX MARTINEAU
DÉVELOPPEUR

WHAT WE DIDN'T COVER

BUT YOU SHOULD KNOW ABOUT

* AMQP (Advanced Message Queuing Protocol)
* Task routing, queue binding
* Broker HA
* Broker security

ALIX MARTINEAU
DÉVELOPPEUR

REFERENCES

- Celery documentation
- RabbitMQ documentation
- Sample Django project with Celery integration
- Celery 4.0 release log

TECH TALK

ANIMÉ PAR

ALIX MARTINEAU

DÉVELOPPEUR

alix.martineau@ripplemotion.fr

ripple motion services

MERCI DE VOTRE PARTICIPATION !

ripple motion services

ripple motion
services

Créateur d’applications mobiles