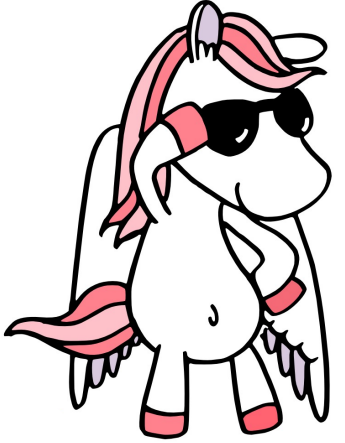


Django et XMPP

Présentation de l'intervenant

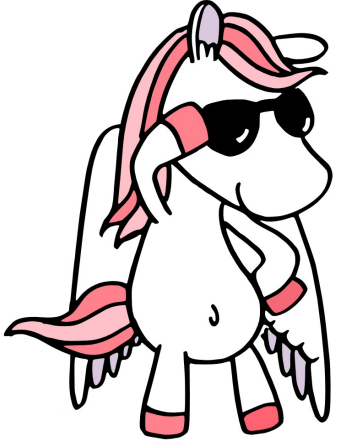
- Jean-Michel ARMAND aka MrJmad

- http://twitter.com/mrjmad
- http://j-mad.com/blog/
- http://hybird.org

Licences

- Le contenu de cette présentation est en licence creative commons Paternité (http://creativecommons.org/licenses/by/2.0/fr/), paternité Jean-Michel ARMAND

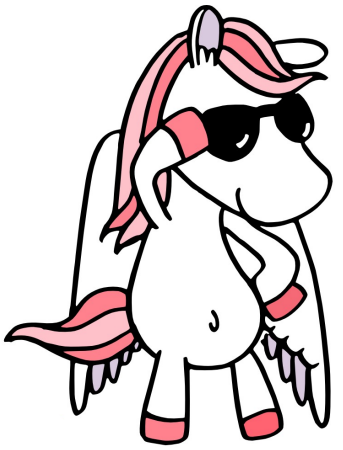
- Le poney utilisé en illustration est lui aussi en licence Creative Commons Paternité, Djangocong en paternité.



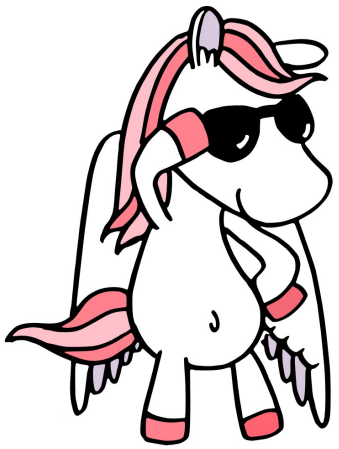
C'est quoi XMPP ?

C'est quoi XMPP ?

- Protocole extensible de message et de présence
- Basé sur une archi client/serveur
- Échange décentralisé de messages instantanés ou non
- En format XML
- Découpé en deux parties :
  - Protocole de base
  - XEP (XMPP Extension Proposal)

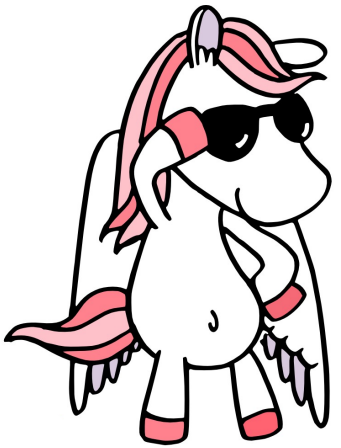


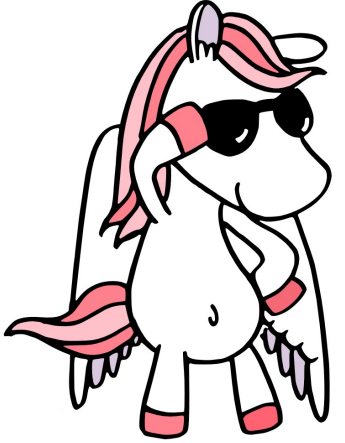
Quand ne pas utiliser XMPP ?



Quand ne pas utiliser XMPP ?

- Pour de la communication inter process, même nominative
- Pour de la délégation de tache avec des producteurs / consommateurs.

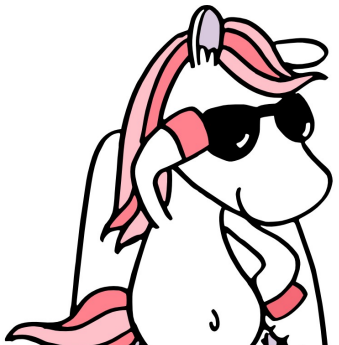




Pourquoi utiliser XMPP ?

Pourquoi utiliser XMPP ?

- Parce que c'est hype
- Parce que c'est le futur du Web
- Parce que c'est stateful, pas comme dans le Web
- Parce qu'on peut faire plein de choses très sympathique.



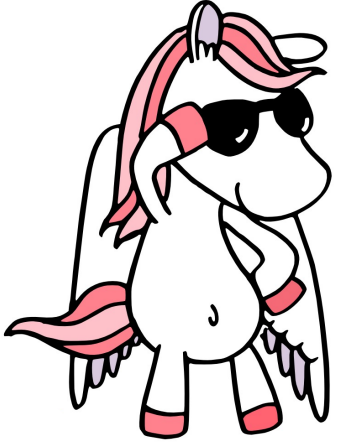
Pourquoi utiliser XMPP ?

- Et sinon....
- Un système de présence et de ressource bien pensé
- Plein de XEP sympa
- Du XML tout partout
- Un mécanisme de déclaration et de découverte de ressource qui peut être utile

Les librairies ?

(ou : c'est la loose, y en a pas et elles sont merdiques...)

Les librairies XMPP



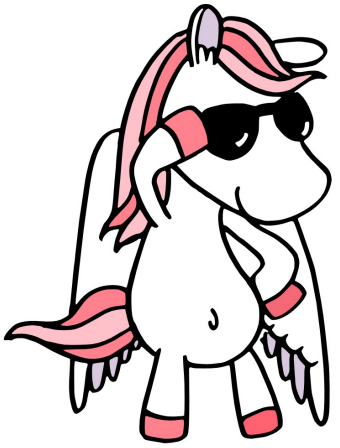
- Twisted Words
- Wokkel
- jabber.py
- Xmmpppy
- PyXMPP
- SleekXMPP
- XMPPony

Les librairies XMPP

- SleekXMPP
  - http://wiki.github.com/fritzy/SleekXMPP
- PyXMPP
  - http://pyxmpp.jajcus.net/

SleekXMPP

- Développement actif (dernier commit il y a trois jours)
- Design par plugin, chaque XEP est un plugin
- Un nombre important de XEP gérée :
  - 4, 9, 30, 45, 50
  - 60, 78, 86, 92, 99



PyXMPP

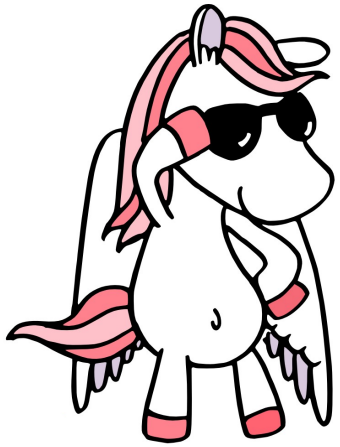
- Développement actif (dernier commit il y a trois jours)
- Pas mal de XEP gérées. (presque autant que SleekXMPP
- Peu de doc hormi l'API.

Twisted et Wokkel

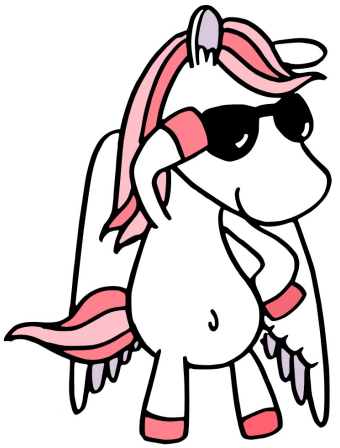
- Ben c'est du twisted quoi donc :
  - Faut embarquer tout twisted
  - Mécanisme des reactor par forcément sympathique dans notre cas d'utilisation
  - Gestion de rien, tout est basé sur les chemins XPATH

Utilisations ?

- Envoyer des statuts à un utilisateur à travers son client de messagerie instantanée
  - Twitter
  - Identi.ca
  - FriendFeed...
- Permettre au django de 'poser des questions' à un admin en push et lui permettre de répondre



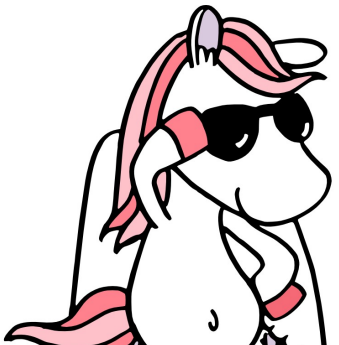
Les XEP cools



XEP-0009: Jabber-RPC

- Du xmlrpc à travers XMPP
- On enregistre des fonctions qui peuvent ensuite être appelées (logique).

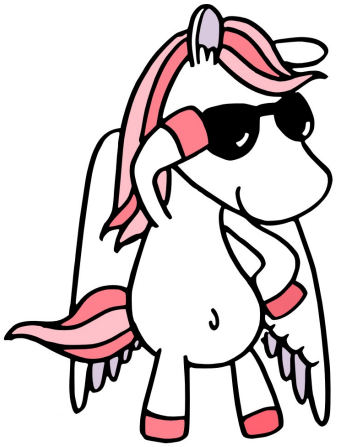
```
<iq type='set'
    from='requester@company-b.com/jrpc-client'
    to='responder@company-a.com/jrpc-server'
    id='rpc1'>
  <query xmlns='jabber:iq:rpc'>
    <methodCall>
      <methodName>examples.getStateName</methodName>
      <params>
        <param>
          <value><i4>6</i4></value>
        </param>
      </params>
    </methodCall>
  </query>
</iq>
```



```
<iq type='result'
    from='responder@company-a.com/jrpc-server'
    to='requester@company-b.com/jrpc-client'
    id='rpc1'>
  <query xmlns='jabber:iq:rpc'>
    <methodResponse>
      <params>
        <param>
          <value><string>Colorado</string></value>
        </param>
      </params>
    </methodResponse>
  </query>
</iq>
```

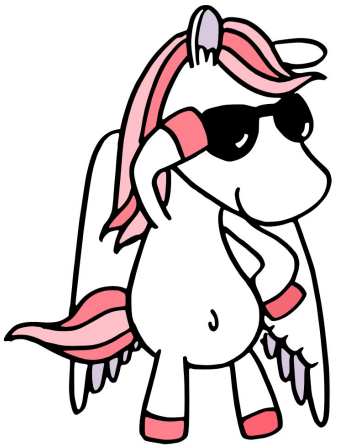
XEP-0009: Utilisation ?

- Faire du xmlrpc ..



XEP-0045: Multi-User Chat

C'est comme IRC, mais en XML



XEP-0045: Utilisation ?

- La discussion, comme sur irc
- Notifier à tout les utilisateurs présents, un message, d'un seul coup.
  - Twitter à base de chan xmpp, follow = joindre un chan

XEP-0060: Publish-Subscribe

- Mécanisme de publication / souscription
- On crée des topics sur lesquels on publie des contenus.
- Les utilisateurs souscrivent aux topics qui les intéressent et sont notifiés quand un nouveau contenu est publié.
- Problème de perf dans les implémentations actuelles

XEP-0060: Utilisation ?

Flux RSS en push et non en pull

XEP-0013: Flexible Offline Message Retrieval

- Comme du POP3, mais pour les messages XMPP
- On peut récupérer soit :
  - Le nombre de messages
  - Les headers des messages
  - Tout
  - Un seul message
- On peut supprimer des messages

XEP-0004: Data Forms

- Comment décrire et envoyer des date à travers XMPP
- Exemple :

```
<x xmlns='jabber:x:data'
   type='{form-type}'>
  <title/>
  <instructions/>
  <field var='field-name'
         type='{field-type}'
         label='description'>
    <desc/>
    <required/>
    <value>field-value</value>
    <option label='option-label'><value>option-value</value></option>
    <option label='option-label'><value>option-value</value></option>
  </field>
</x>
```

Ca ressemble pas à un truc ?

XEP-0050: Ad-Hoc Commands

- Permet de déclarer des commandes qui pourront être lancé par des users authentifiés.
- Les commandes peuvent recevoir en argument des data forms
- Les commandes peuvent renvoyer des dataforms.
- On peut chainer les commandes (comme un wizard) avec des actions (prev, next, cancel, complete)

Deux manières de penser les choses

- Django en temps que client XMPP
  - Comme un autre client, il se connecte à un serveur et échange avec d'autres utilisateur
  - Simplicité de la chose
  - On se limite aux XEP gérées par le serveur
- Django en temps que serveur
  - Les utilisateurs se connectent directement en XMPP à Django
  - Peu de lib prévu pour
  - Gros boulot à faire

Django en temps que client

- Des connexions déconnexion non stop pour chaque requête
  - BOUHHHH
- Une commande django qui se lance à coté, communique avec le vrai django (passage de message timestampé en BD?)
  - OUAIISSSSS

Merci de votre attention