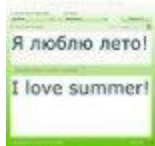
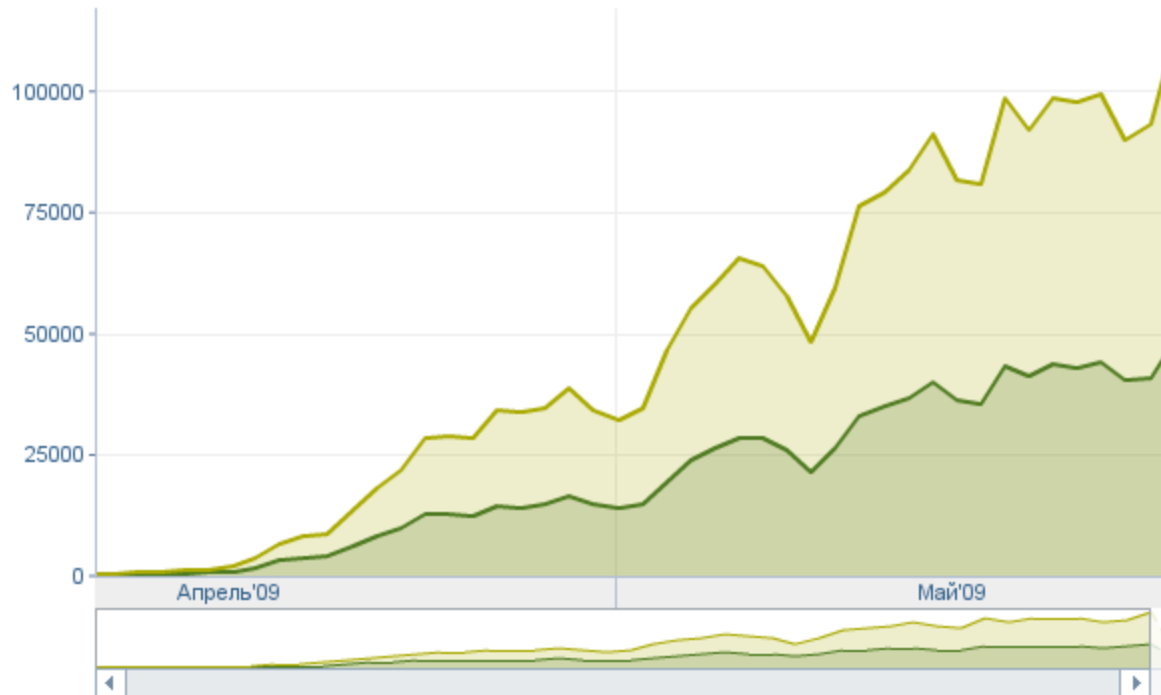


# Технологии которые могут помочь создать социальное приложение.



Уникальные посетители и просмотры



- просмотры
- уникальные посетители



# Кто я?

- Юра Безнос
- System administration linux с 2002 года.
- Web Designer
- Программист as3/php/perl/python/java и т.п.
- Предприниматель с весны 2010г.

Материалы выступления:

**<http://blog.you-ra.info/>**

# Зачем технологии?

Идея

Прототип

???



# Инструменты

Клиенская часть:




- Flex/Flash P2P
- Google Web Toolkit(GWT)

Серверная:

- Tornado(web server)+NginX
- Twisted(networking engine)



# Flash P2P

The Evolution of Media and Communication Delivery on the Flash Platform		
Traditional Streaming / Communication with Unicast model	RTMFP in Flash player 10.0 / Cirrus 1	RTMFP in Flash player 10.1 / Cirrus 2
		
Traditional streaming requires a client to receive all data from a centralized server cluster. Scale is achieved by adding more servers	First generation of RTMFP in Flash player 10.0 supported rendezvous. Media was always sourced from the publishing peer.	Second generation of RTMFP supporting groups in Flash player 10.1 supports application-level multicast and reduces the load on the source publisher.

# Flash P2P libraries



- **P2PGameEngineLibrary(MAX2010)**
- <http://www.flashrealtime.com/p2p-game-lib/>



- **HydraP2P**
- <https://github.com/devboy/HydraP2P>
- <http://devboy.org/2010/09/07/hydrap2p-own-commands/>



# Google Web Toolkit



- GWT SDK
- Java API libraries
- Compiler
- Development server



- Plugin for Eclipse
- IDE support for GWT



- GWT Designer



# GWT Designer

The screenshot displays the GWT Designer IDE interface. On the left, the **Structure** pane shows a project named "Application.java" with a tree view of components: "shell - 'Application'", "nameComposite", "Address", "Phone", "home - 'Home'", "homePh - ''", "office - 'Office'", "officePh", "mobile - 'Mobile'", "mobilePh", and "emailComposite". Below this is the **Properties** pane for the selected "home" component, showing variables and style properties.

The **Palette** pane in the center lists various widgets and layouts, including System, Selection, Marquee, Composite, Canvas, Table, Tree, ListBox, ListView, SortedList, Absolute layout, FillLayout, GridLayout, FormLayout, RowLayout, Controls, Push Button, Check Box, Radio Button, Label, Text, Combo, List, ProgressBar, Slider, Browser, Table, TableColumn, TableItem, Tree, Menu, Menu Bar, Popup Menu, Cascade M..., MenuItem, Radio Men..., Check Men..., and Separator.

The main design view on the right shows a form with fields for "Last Name", "Street", "City", "State", and "Zip". Below these is a table with columns for "Home" and "Office", and rows for "Mobile" and "Email". A language selection dropdown menu is open, showing options for "(default)", "de", "en\_US", and "ru".

At the bottom right, the text "WindowBuilder Pro" and "www.windowbuilderpro.com" is visible.





# Tornado

- non-blocking and uses epoll
- can handle 1000s of simultaneous standing connections
- ideal for real-time web services

## Modules:

- web - The web framework on which FriendFeed is built. web incorporates most of the important features of Tornado
- escape - XHTML, JSON, and URL encoding/decoding methods
- database - A simple wrapper around MySQLdb to make MySQL easier to use
- template - A Python-based web templating language
- httpclient - A non-blocking HTTP client designed to work with web and httpserver
- auth - Implementation of third party authentication and authorization schemes (Google OpenID/OAuth, Facebook Platform, Yahoo BBAuth, FriendFeed OpenID/OAuth, Twitter OAuth)
- locale - Localization/translation support
- options - Command line and config file parsing, optimized for server environments

# Twisted is an event-driven networking engine

- Twisted projects variously support TCP, UDP, SSL/TLS, multicast, Unix sockets, a large number of protocols (including HTTP, NNTP, IMAP, SSH, IRC, FTP, and others), and much more.

```
from twisted.internet.protocol import Protocol, Factory
from twisted.internet import reactor
```

```
class Echo(Protocol):
    def dataReceived(self, data):
        """
        As soon as any data is received, write it back.
        """
        self.transport.write(data)
```

```
def main():
    f = Factory()
    f.protocol = Echo
    reactor.listenTCP(8000, f)
    reactor.run()
```

Спасибо!