TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

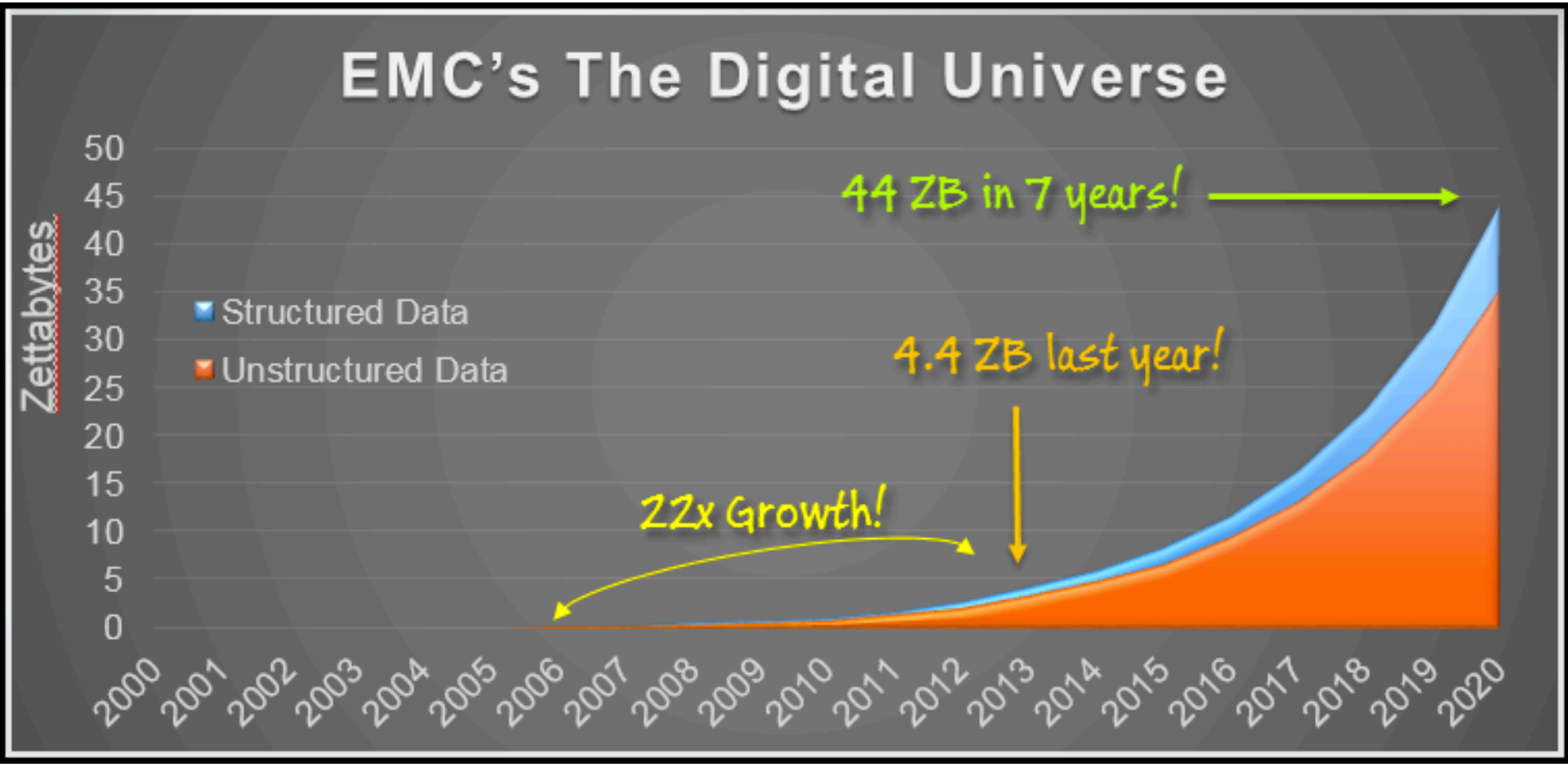
Getting the data
Web crawling

Viet-Trung Tran
School of Information and Communication Technology

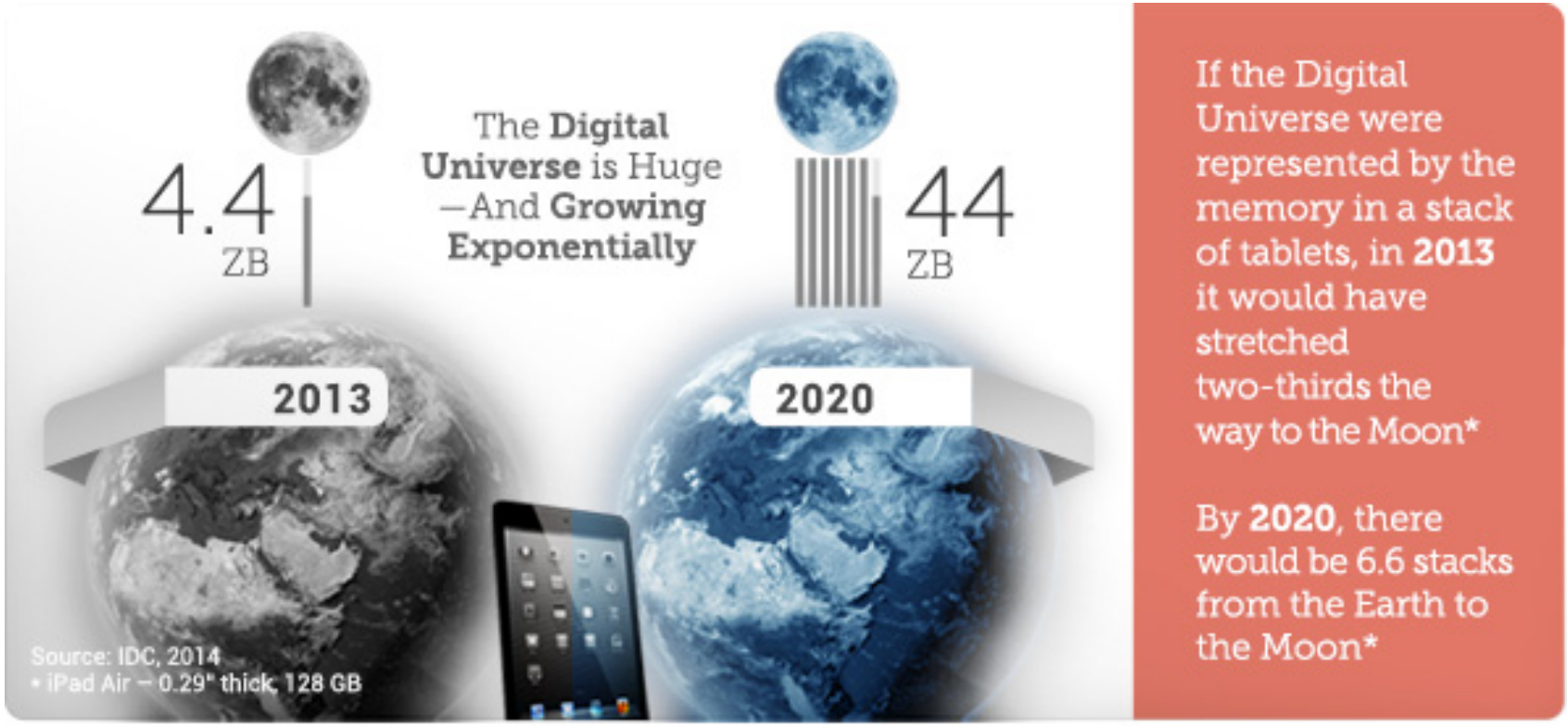
Outline

- Introduction
- Internet crawlers
- Scrapy framework
- Facebook graph API

How big is our digital universe

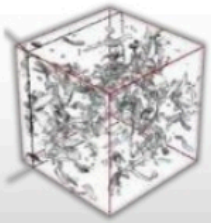


How big is big data?



Data science: The 4th paradigm for scientific discovery

$$\left(\frac{\dot{a}}{a}\right)^2 = \frac{4\pi G\rho}{3} - \mathrm{K}\frac{c^2}{a^2}$$



Experimental	Theoretical	Computational	The Fourth Paradigm
Thousand years ago	Last few hundred years	Last few decades	Today and the Future
Description of natural phenomena	*Newton's laws, Maxwell's equations...*	*Simulation of complex phenomena*	*Unify theory, experiment and simulation with large multidisciplinary Data* *Using data exploration and data mining (from instruments, sensors, humans...)* *Distributed Communities*

Big data in 2008

http://www.wired.com/wired/issue/16-07

September 2008

Big data in 2014

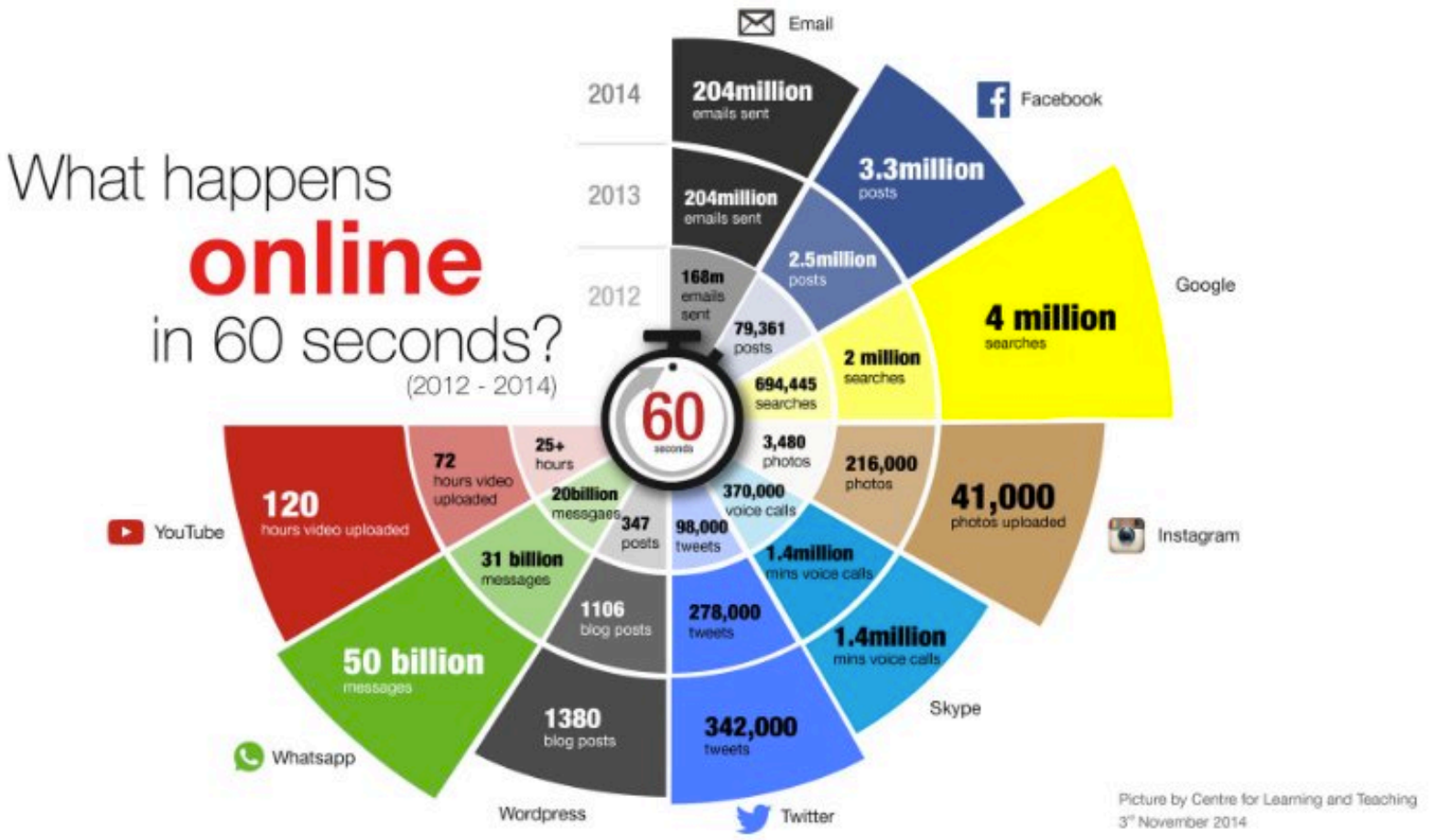
THE AVERAGE PERSON TODAY PROCESSES MORE DATA IN A SINGLE DAY THAN A PERSON IN THE 1500'S DID IN AN ENTIRE LIFETIME

LOOK TO THE LEFT, and you see Times Square at dusk. Look to the right, and you see the same location at midmorning. Internationally acclaimed photographer Stephen Wilkes's time-altering image of New York's Times Square is part of his body of work titled *Day to Night*. The image was created by blending more than 1,400 separate photos taken over the course of 15 hours—a meticulous process that took him nearly three months.

PHOTO: STEPHEN WILKES

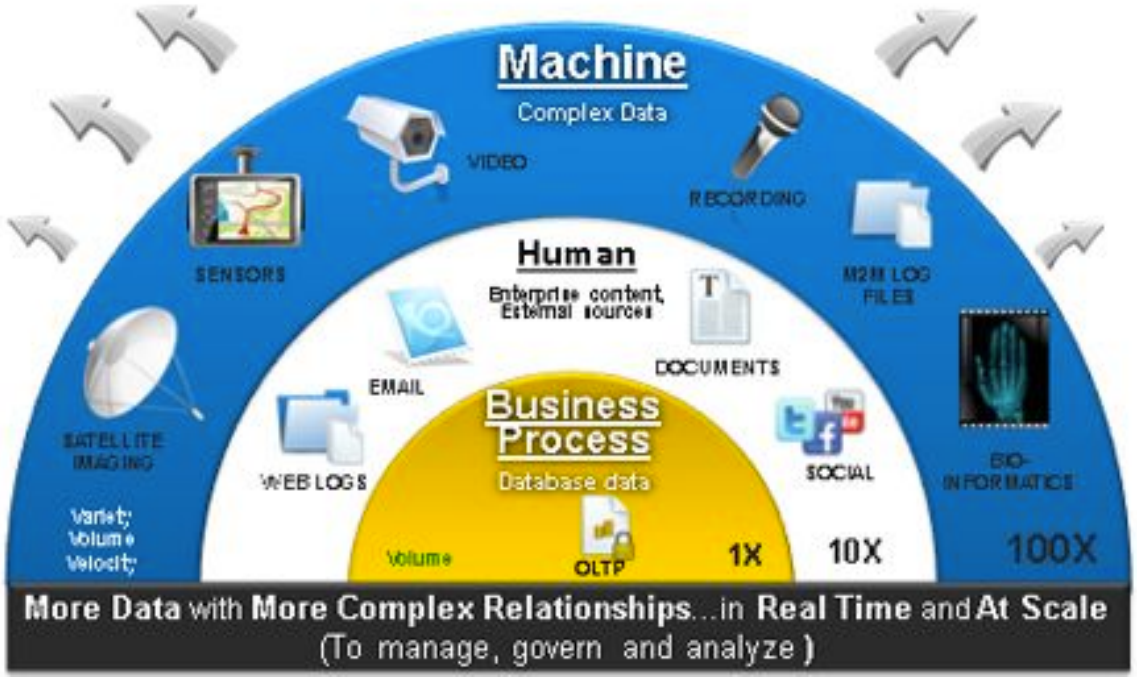
Big data today

Big data's numbers



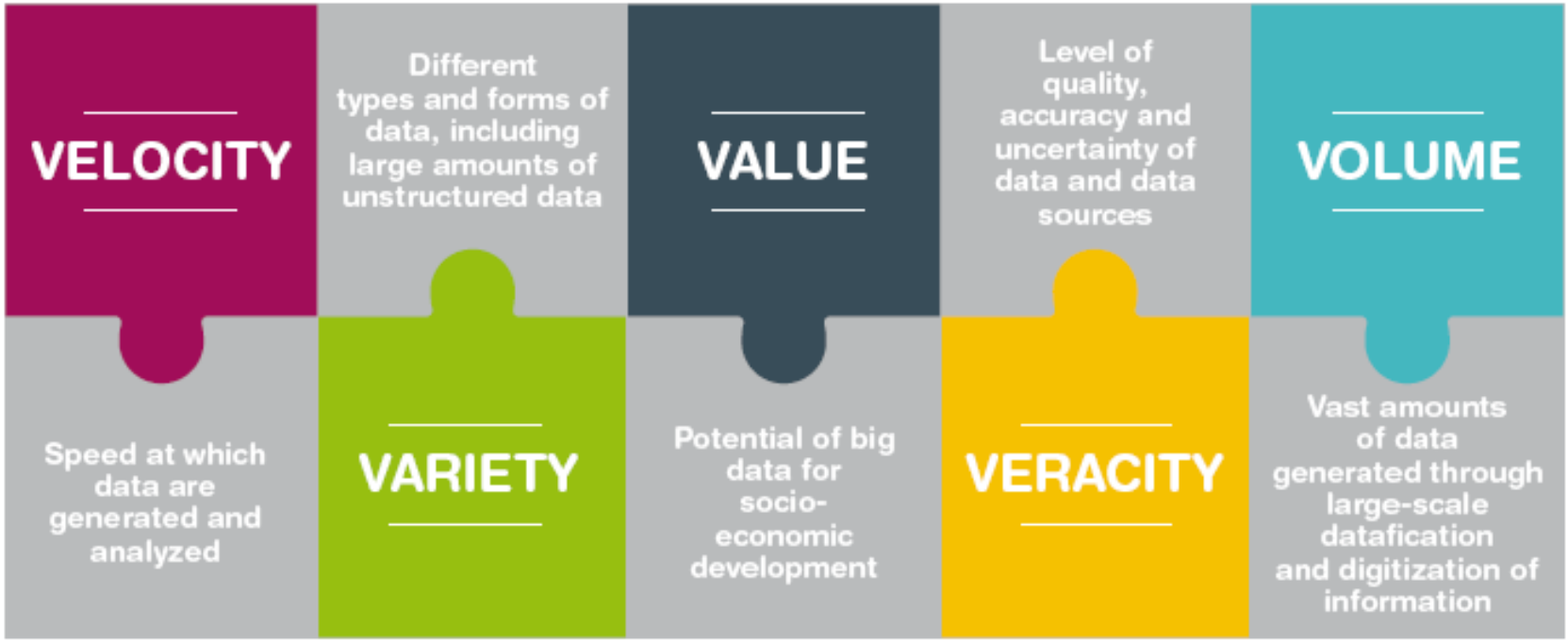
Big data sources

- E-commerce
- Social networks
- Internet of things
- Data-intensive experiments (bioinformatics, quantum physics, etc)



Data is the new oil

Big data 5'V



Big data is a term for data sets that are so large or complex that traditional data processing application software is inadequate to deal with them (wikipedia)

Internet crawlers

To collect dat from Internet into your database

Web crawling

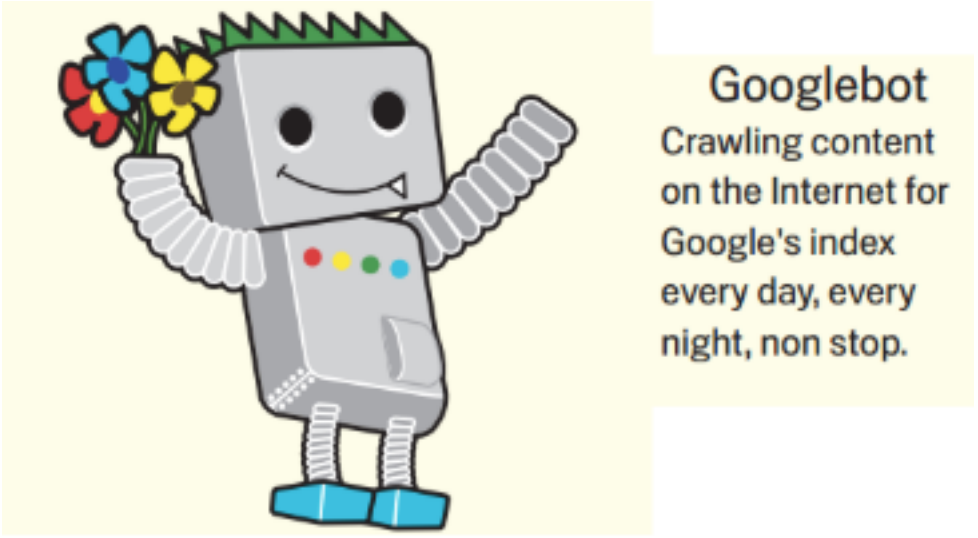
Public APIs

Objectives of Web crawling

- Universal search engines
  - Google, Bing, CocCoc, etc
- Vertical (specialized) search engines
  - News, real-estate, reviews, shopping, etc.
- Social media analysis
  - Community detection, trending topics, sentiment analysis, key influencers
- Business intelligence
  - Competitor/client analysis
- All sort of stuffs
  - Email, phone numbers, personal photos, etc. ☹

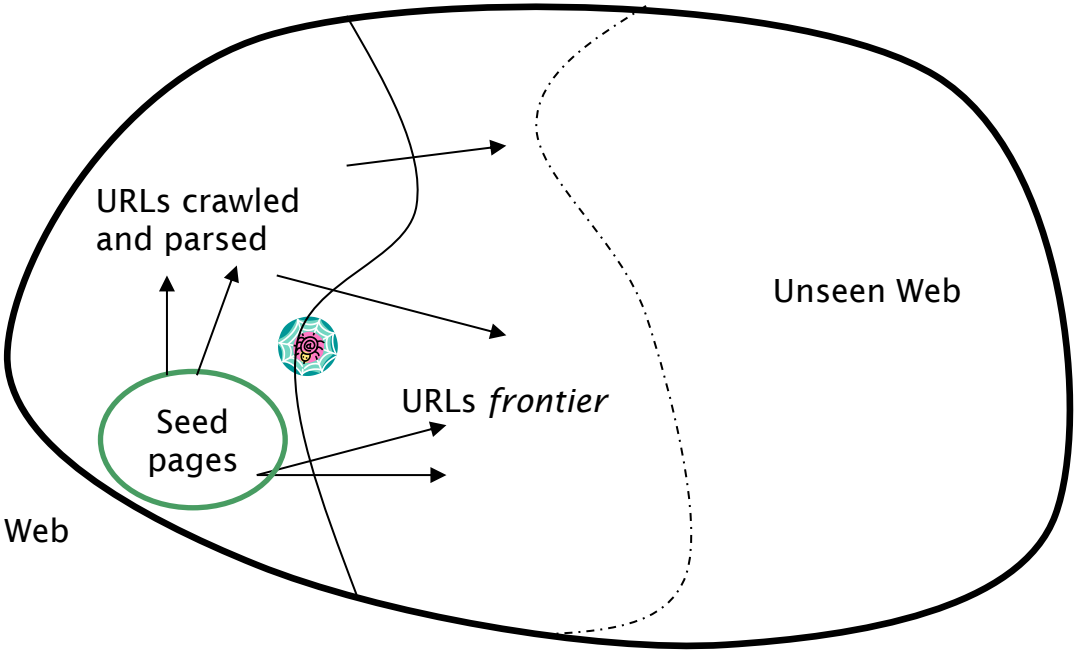
Web crawlers in many names

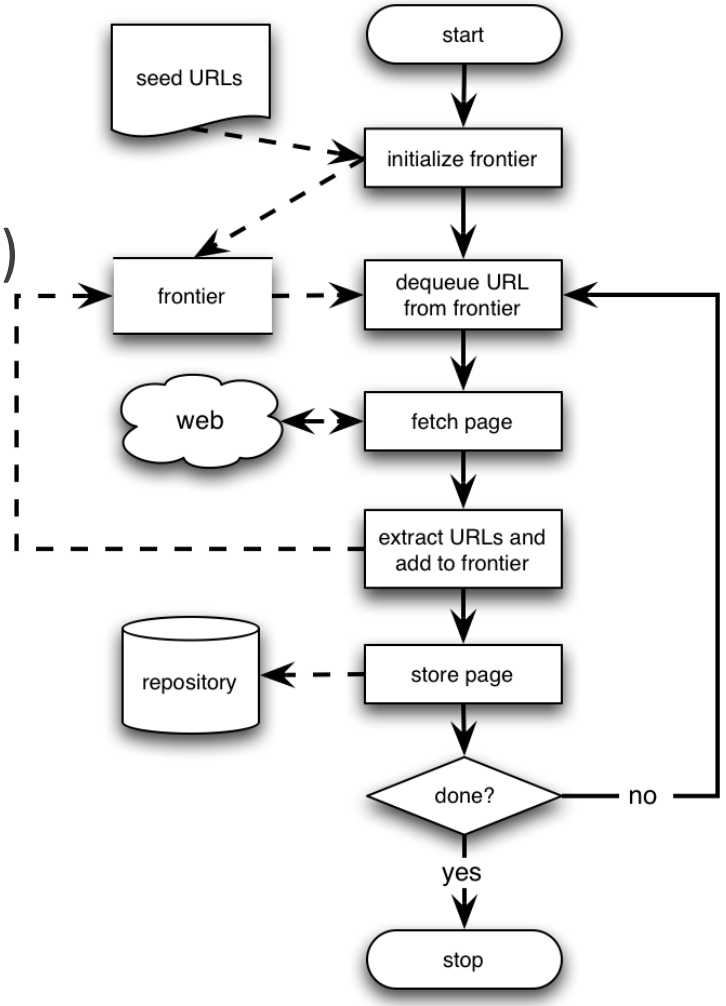
- A Spider, also known as a robot or a crawler, is actually a program that follows, or ”crawls” links throughout the Internet, grabbing content from sites and adding it to the database.
  - Crawler
  - Spider
  - Robot
  - Web agent



Basic crawler operation

- Begin with known “seed” URLs
- Fetch and parse them
  - Extract URLs they point to
  - Place the extracted URLs on a queue
- Fetch each URL on the frontier (queue) and repeat





Web crawler types

- Incremental crawler
- Distributed crawler
- Universal crawler
  - For universal search engines
  - Large-scale
  - Huge cost
  - Incremental updates
- **Focused crawler (we are more focused on this type)**
  - Focus on a particular, specialized data

Challenges

- The Internet is huge
  - Googlebot are distributed
- Filtering interested/non-interested/malicious pages
  - Spam pages
  - Spider traps – pages that are dynamically generated
- Content freshness
  - Crawlers should be catchup with new, up-to-date contents
- Content deduplication
  - Site mirrors and duplicate pages
- Politeness – don’t hit a server too often

Trading-off exploitation vs. exploration

- Exploitation
  - the crawling of pages where the expected value can be predicted with a high confidence
- Exploration
  - the search for new sources of relevant pages

Politeness

- Explicit
  - Specified by webmasters on which parts of the site can be crawled (robots.txt)
- Implicit
  - Avoid hitting any site too often to consume too much webserver resource

Robots.txt

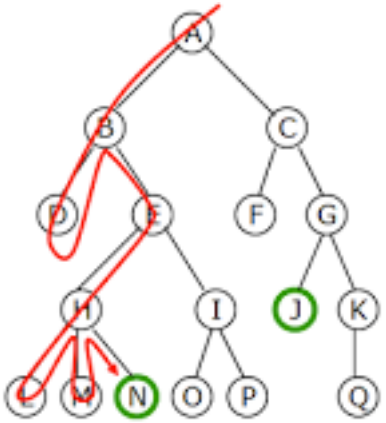
- Protocol for giving spiders "robots" limited access to a website, originally from 1994
  - www.robotstxt.org/wc/norobots.html
- Website announces its request on what can(not) be crawled
  - For a server, create a file `/robots.txt`
  - This file specifies access restrictions
- Example

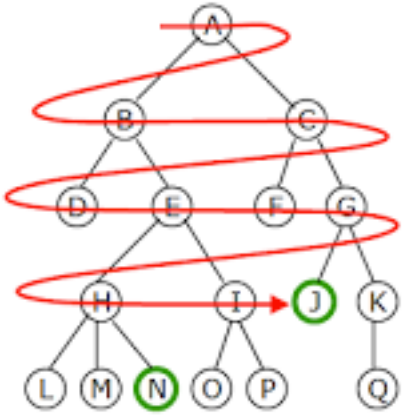
```
User-agent: *
Disallow: /yoursite/temp/

User-agent: searchengine
Disallow:
```

Focused crawler main tasks

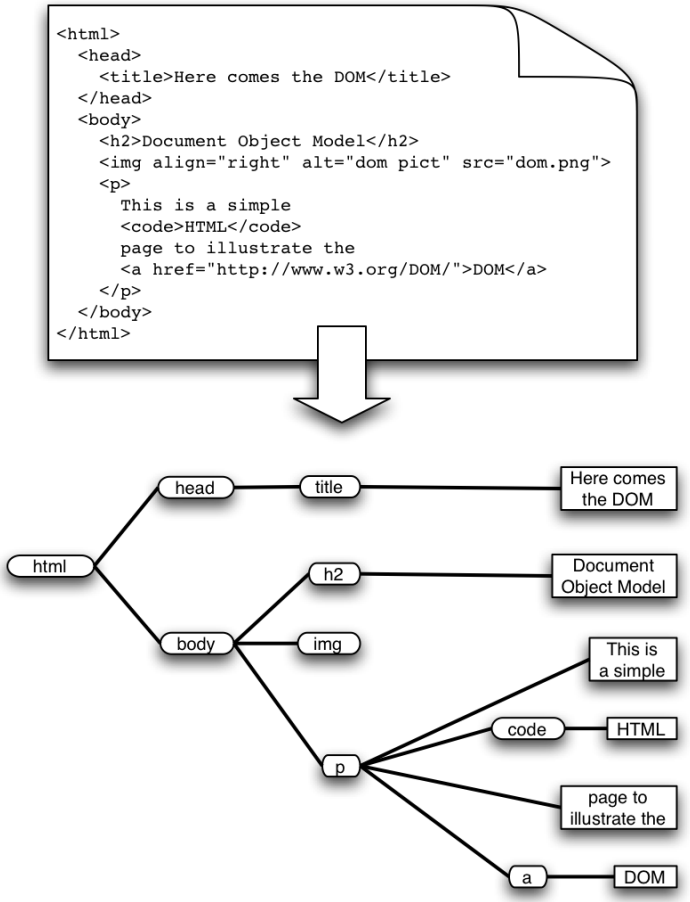
- Which to crawl
  - Is this site worth to visit?
  - Is this URL worth to crawl?
- How to crawl
  - Visit/Revisit strategies
- How to parse/extract data autonomously
  - Dynamic wrappers (Natural language processing)



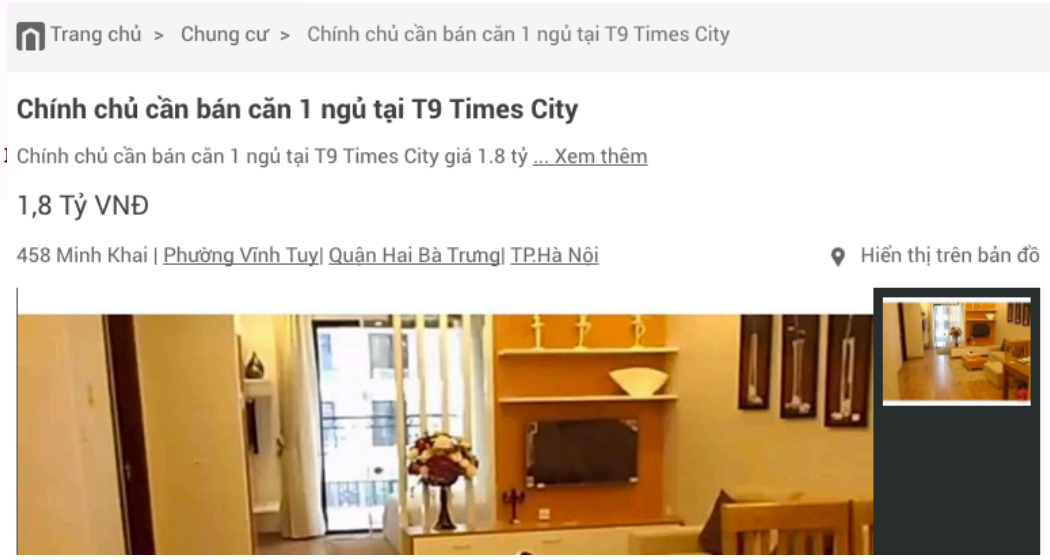


Parse HTML content

- HTML has the structure of a Document Object Model tree
- DOM tree varies from page to page, even for the same catalog
  - Due to dynamic contents, dynamic ads, etc.



Dynamic wrappers



Implementation issues

- Don’t want to fetch same page twice!
  - Keep lookup table (hash) of visited pages.
  - What if not visited but in frontier already?
- The frontier grows very fast!
  - May need to prioritize for large crawls
- Fetcher must be robust!
  - Don’t crash if download fails, timeout mechanism
  - Detect and break redirection loops
- Determine file type to skip unwanted files
  - Can try using extensions, but not reliable
  - Can issue ‘HEAD’ HTTP commands to get Content-Type (MIME) headers, but overhead of extra Internet requests

Tools and framework for crawling

General Features Comparison					
	Octoparse	**Parsehub**	**Mozenda**	**Dexi.io**	**Import.io**
Usability	★★★★★	★★★★☆	★★★★★	★★★★★	★★★★☆
Functionality	★★★★☆	★★★★☆	★★★★☆	★★★★★	★★★☆☆
Easy to learn	★★★★★	★★★★☆	★★★★★	★★★☆☆	★★★★★
Customer support	Email, phone, community	Email, live chat, forum	Phone, email, video chat	Email, phone, community	Email, chat bot, community
Price	$0 - $249	$149 - $499	$100/5000 page credits	$119 - $699	$299 - $9999
Trial/Free version	Free Version	Free Version	30 days trial	Trial	7 days trial
OS (Specifications)	Win	Win, Mac, Linux	Win	Win, Mac, Linux	Win, Mac, Linux
Data Export Formats	TXT, CSV, XLS, Databases	CSV, JSON	CSV, TSV, XML, XLS, JSON	CSV, XLS, XML, JSON, Zip	CSV, JSON, Google sheets
Multi-thread	✔	✔	✔	✔	✗
API	✔	✔	✔	✔	✔
Scheduling	✔	✔	✔	✔	✔

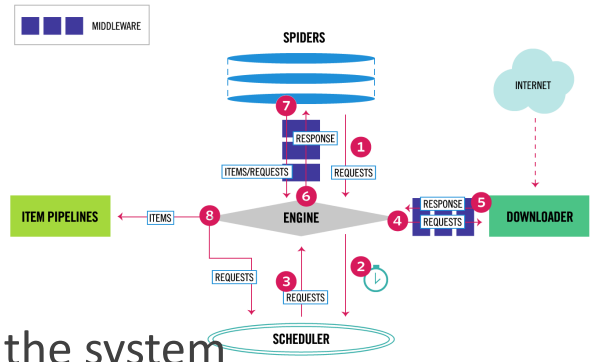
Scrapy

- Python
- Open source web scraping framework
- Scrap websites and extract structured data

Why use Scrapy

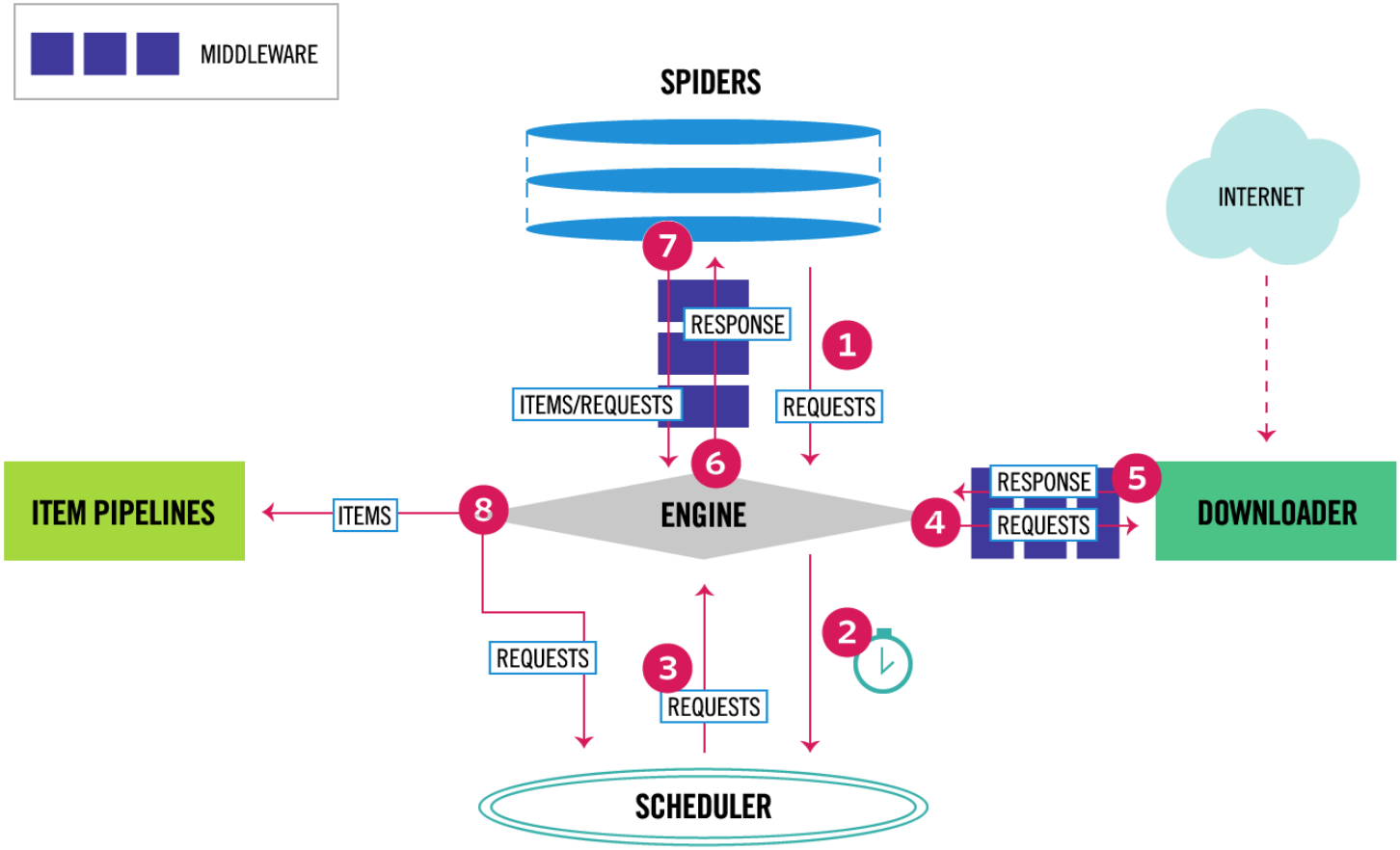
- Annoying stuffs in crawling and scraping are done by Scrapy
  - Extracting links
  - Throttling
  - Concurrency
  - Robots.txt and <meta> tags
  - XML sitemaps
  - Filtering duplicated URLs
  - Retry on Error

Scrapy components



- Scrapy Engine
  - controlling the data flow between all components of the system
- Scheduler
  - receives requests from the engine and enqueues them for feeding them later
- Downloader
  - fetching web pages and feeding them to the engine
- Spiders
  - parse responses and extract items or additional requests to follow
- Item pipeline
  - processing the items once they have been extracted by the spiders
- Downloader middlewares
  - process requests when they pass from the Engine to the Downloader and vice-versa
- Spider middlewares
  - specific hooks that sit between the Engine and the Spiders
  - process spider input (responses) and output (items and request

Scrapy architecture



Working with scrapy

- https://doc.scrapy.org/en/latest/intro/tutorial.html
- Define your own data structures
- Write spider
  - Leverage Built-in Xpath and CSS selectors to extract desired data
  - Built-in Json, csv, xml output
- Interactive shell console

Downloading and processing files and images

- https://doc.scrapy.org/en/latest/topics/media-pipeline.html#topics-media-pipeline

Facebook graph API

Graph API

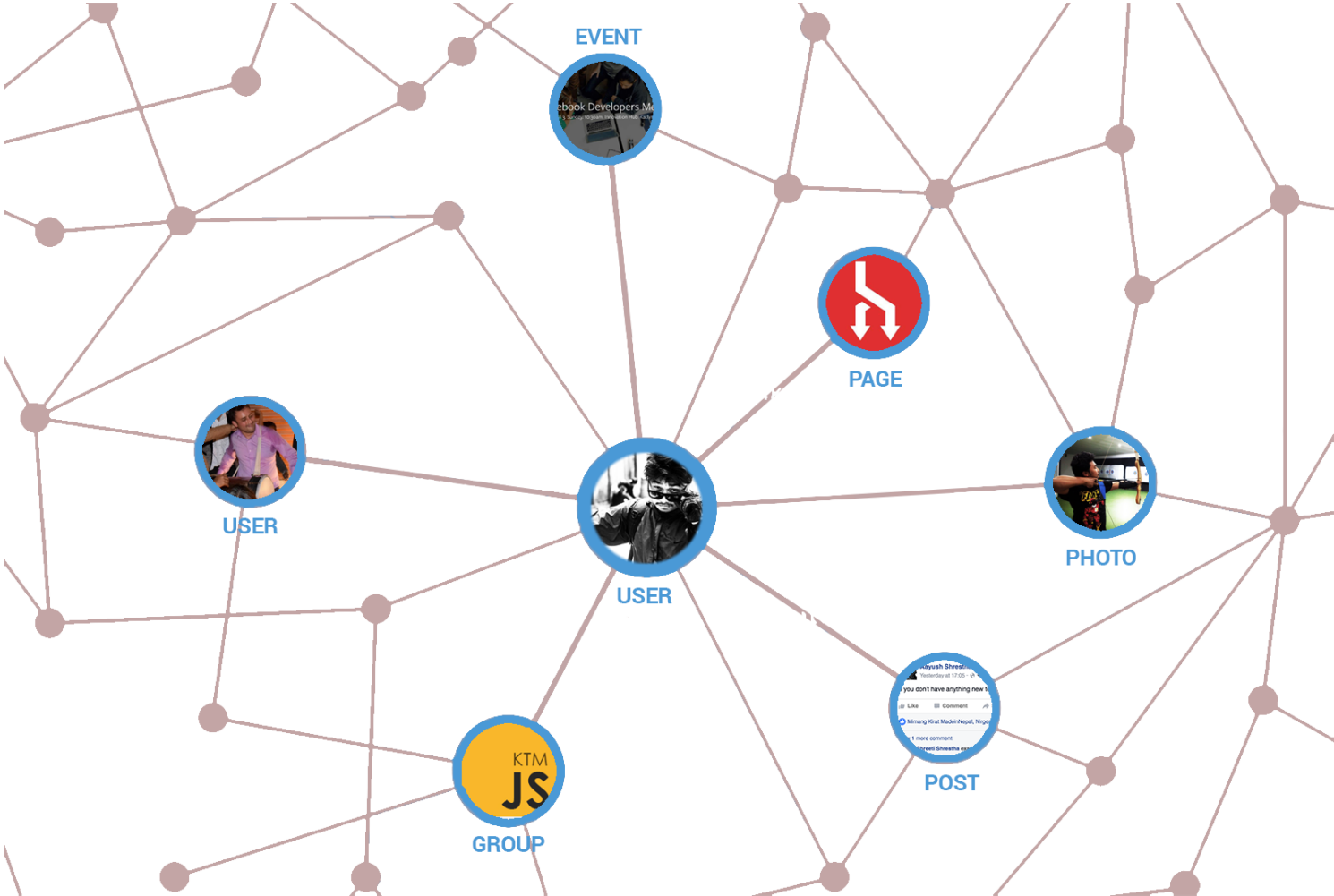
- Primary way for apps to get data in and out of the Facebook Social Graph
- HTTP Based REST API
- You can
  - query data
  - post status and stories
  - upload pictures and videos and more …

Social graph

- Representation of the information on Facebook
  - Nodes
  - Edges
  - Fields

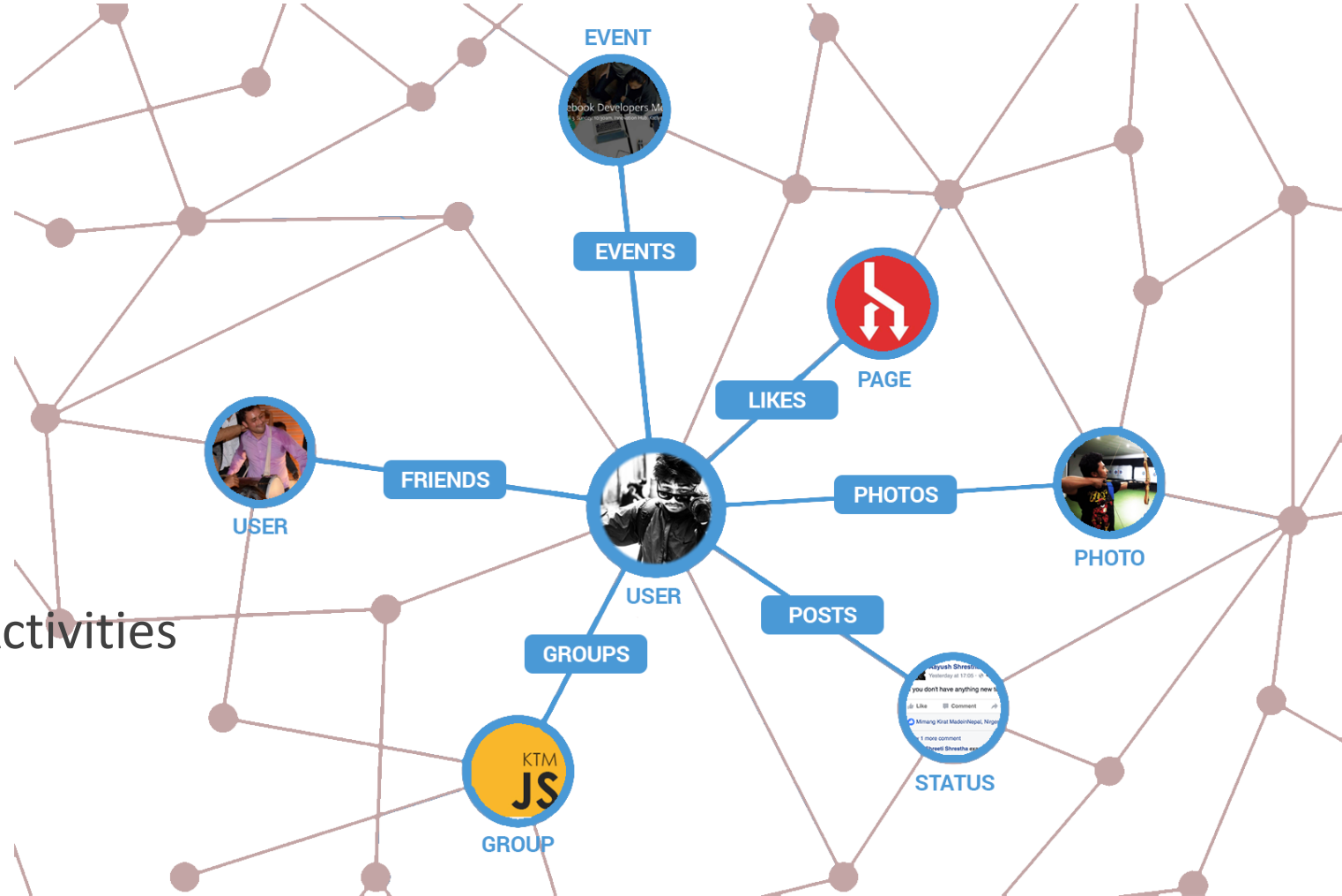
Social graph - nodes

- Every “thing”, such as a user, a photo, a comment, a page is a node
- User
- Photo
- Album
- Event
- Group
- Comment
- Story
- Video
- Link
- Note
- Page



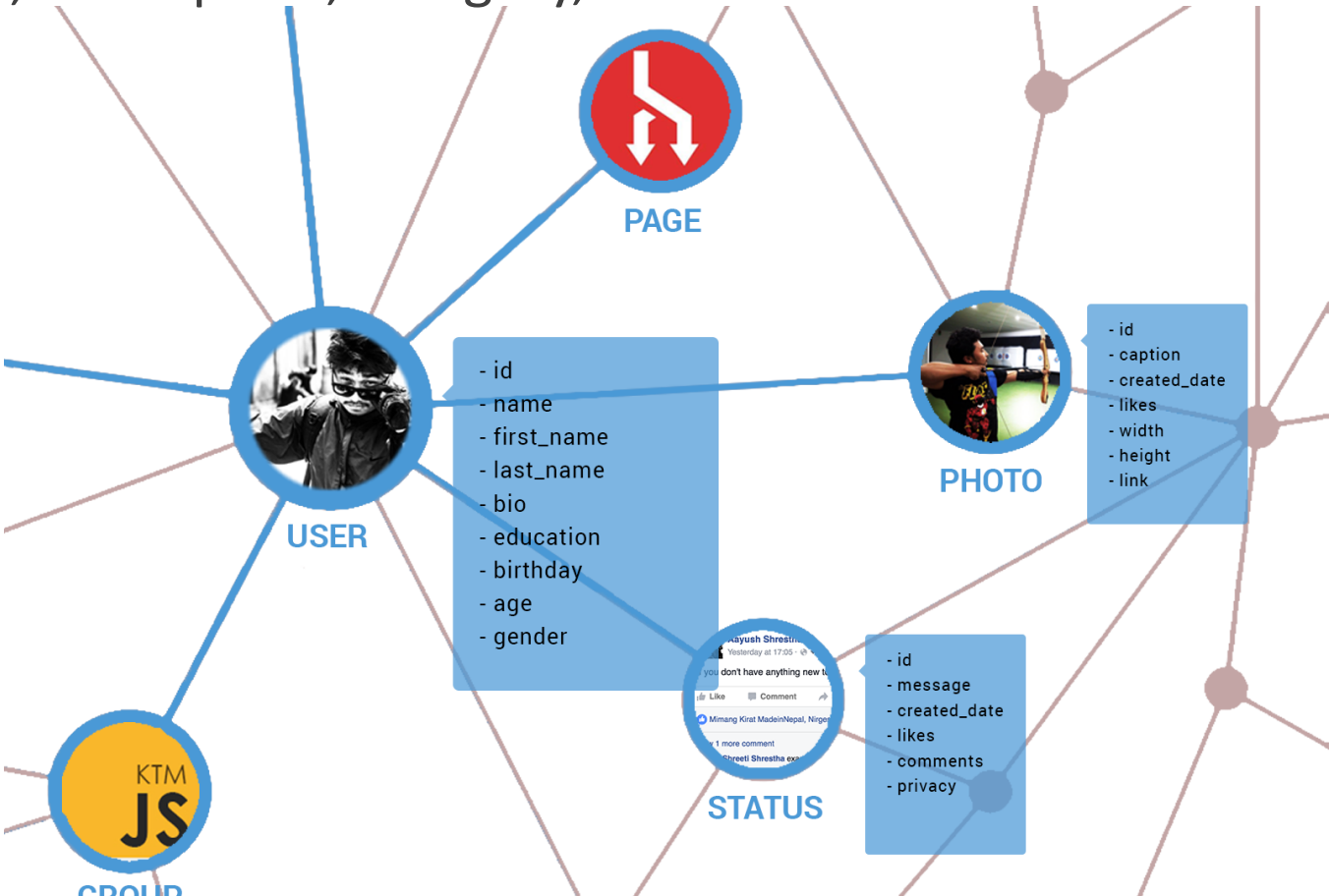
Social graph - edges

- The connection between these “things” are edges.
- Feed
- Tagged
- Posts
- Picture
- Interests
- Likes
- Photos
- Statuses
- Friends Activities



Social graph - fields

- The information about these “things” are fields.
- User has : name, age, birthday, etc.
- Page has : name, description, category, etc



Access token

- Making Graph API calls require an **Access Token**
- Get your Access Token using **Facebook Login** or one of the **SDK's**
- Get familiar with the **Graph API Explorer** http://developers.facebook.com/tools/explorer

Traversing through the graph

- Authentication : OAuth 2.0
- Selection (or Query)
- Basic Operations (Publishing, Updating and Deleting)
- Searching
- Introspection

Selection - Selecting nodes

- graph.facebook.com/{node_id}
- graph.facebook.com/{node_username}
- graph.facebook.com/{node_id}?fields=id,name
- **TRY THESE**
  - 1. /me?field=id,name
  - 2. /me?field=album.limit(10){name,likes,count},photos
  - 3. /album_id
  - 4. /page_id

Selection – Selecting collections

- graph.facebook.com/{node_id}?fields={connection_name}
- graph.facebook.com/{node_id}/{connection_name}
- graph.facebook.com/{node_id}/{connection_name}?fields=id,name
- **TRY THESE**
  - 1. /me/friends
  - 2. /me/friends/friend_id
  - 3. /albums
  - 4. /photos?type=uploaded

Publishing

- Publishing is done in **edges**
  - graph.facebook.com/{node_id}/{connection_name} - POST Request
- **TRY THESE**
  - 1./me/feed - Fields : message=Hello World!
  - 2./me/feed - Fields : message=Hello World!, privacy = {value : ‘SELF’}

Updating

- Make **POST** Requests, now on nodes
  - graph.facebook.com/{node_id} - POST Request
- **TRY THESE**
  - 1. /{node_id}-Fields:message=HelloWorldAgain!!

Deleting

- Make **DELETE** Request on the node
  - graph.facebook.com/{node_id} - DELETE Request
- **TRY THESE**
  - 1./{story_id}

Searching

- graph.facebook.com/search
- **TRY THESE**
  - 1. /search?q=john&type=user
  - 2. /search?q=facebook+meetup&type=event
  - 3. /search?q=coffee&type=place&center={lat},{lon}&distance=1000

Introspection

- graph.facebook.com/{node_id}?metadata=1
- JSON comes with the metadata of the node
- What type of node is this?
- What are its fields and what do they represent?
- What connections does this node possess?
- **TRY THESE**
  - 1./me?metadata=1