Building Social Networks

NYC Camp

July 21 2012

#nyccamp
@careernerd & @thinkdropNYC

Jon Pugh
Founder & CTO, ThinkDrop Consulting
thinkdrop.net drupal.org/user/17028

THINK DROP

I NY

Jonathan Pugh

jon@thinkdrop.net
twitter.com/careernerd
Jon Pugh on drupal.org

Professional Web Developer since 1999
Exclusively Drupal since 2004

ThinkDrop Consulting

http://thinkdrop.net
twitter.com/thinkdropNYC
facebook.com/thinkdrop
Now on Google+!

Clients have included Institute for Integrative Nutrition, BlogHer, Sony Music, Imbee.com, & FoodPop.com

We help organizations learn and leverage Drupal.

from Brooklyn New York

WARNING:

Some of the modules you are about to see may be innappropriate for some users (and most developers).

WARNING:

If you see something you don't like, please write a patch, a workaround, or just deal with it.

DO NOT complain to the module contributors. They work for free.

But seriously, a patch would be great...

WARNING:

Building a social network is a challenging, demanding, and often frustrating experience. Your goals should be modest. Don't ever agree to build a feature that works "like Facebook does" without thorough research.

in other words...

PROCEED WITH CAUTION

What is a social network?

*Define **network**?*

A group or system of interconnected people or things

*Define **social** network?*

A network of ***social interactions*** and ***personal relationships***

A dedicated website or other application that enables users to communicate with each other by posting information, comments, messages, images, etc

Every Drupal site is a network, but its social tools are limited.

What does it take to be a social network?

Relationships

Friends, Followers, Circles...
Choose something.
(But don't do everything.)

Communication

Public & Private.
"Write on my Wall"
"Send me a message"
"Mention Me"
"Share This"

Distribution

Get content to interested viewers.
(but not too much content!)
Let users subscribe to content.

Activity

"Joe, Jane, and Pat changed their profile pictures."
*"3 of your friends liked **kittens**"*
"Sam is now friends with Alex and Bob."

Integration

Don't build a walled garden.

Privacy & Permissions

Be thoughtful.
Let users have some control, but don't overwhelm them.

How do they do it?

	Facebook	Twitter	LinkedIn	Drupal Core
Relationships	Friends Networks Fans (now Likes)	Following	Complex Connections	None.
Communication	The Wall Comments The "Share"	Open Stream Direct Messages Mentions Retweet	Exclusive Limited Network	comment.module contact.module
Distribution	News Feed Content and Activity by Friends	Following	Copy of News Feed	/node /taxonomy/term/%
Privacy & Permissions	Friends, Friends of Friends, Everyone Per Post Permissions	Public or Private Accounts		"access content" "access user profiles"
Integration	Applications Platform Connect Social Plugins Open Graph	API First Open API Open Philosophy	Applications Platform	There's a module for that.

How do ***we*** do it?

Relationships	**Communication**	**Distribution**	**Privacy & Permissions**	**Integration**
user_relationships	privatemsg	search_api	flag_friend_access	twitter
flag_friend	dxmpp	search_api_solr	user_relationships_access	rpx
og	references	search_api_views	spaces_permissions	services
relation		entity		views_datasource

Oh, and these...

addressfield admin_menu advanced_help apachesolr apc backup_migrate backup_migrate_files captcha coder commentaccess comment_notify compact_forms

contact_importer context ctools date devel diff ds dummyimage dxmpp entity entity cache
eva extlink fbconnect features feeds field_group fivestar flag flag_friend galleryfor matter
google_analytics job_scheduler jquery_ui lexicon libraries link logintoboggan memcac he
messaging mimemail module_filter multiform nodereference_url oauth og panels
path_alias_xt pathauto plupload porterstemmer privatemsg profile2 quicktabs realna me
references remember_me rpx rules search_api search_api_context

Best Practices

Use them or lose them.

Best Practices!!!

Complex Stories, Complex Testing

Social Networks are more complex than most sites because you have to imagine multiple users interacting with the same content with different configurations while imagining the data and your code simultaneously.

In an ideal world, when all best practices are in place, developing a complex platform can actually be fun.

Higher Expectations

Social Networks are more sensitive to uptime and bug-free functionality than most sites, since they usually cater to some innate human need and are therefor used much more than most websites.

Also, Facebook and Twitter spend millions to make it so

Best Practices!!!

Exportable.

If you can't "export", you can't safely and easily deploy. If its not in code, its volatile, and you can't go back. EXPORT and COMMIT!
See http://drupal.org/project/features and http://drupal.org/project/ctools

Testable.

Run tests. Build tests. Learn test-driven development.
Or crying will become a part of your debugging experience and you will lose more of your life to clicking than you would like to keep track of.
See http://drupal.org/simpletest

Programmable, not configurable.

Rules.module: BAD. PHP Input Filter: VERY BAD. Any PHP you input through a web browser, even in views: BAD BAD BAD!. Drupal Hooks: good. Rules uses hooks to trigger "Rules". You should too. If you can build a "Rule", you can write a module.

Best Practices!!!

One "Drupal" Development Site, localhost for code only.

Features and exportables can be troublesome when developers are passing around exported code and databases from and to a development site and localhost. An old view in a developers database could be exported as new when rebuilding a feature module.

Use ONE central site for ALL Views, Pages, Configuration, Content types, Fields development, and other Drupal web-configurable systems.

EXPORT features from the Dev site to code, but the dev site is the bleeding edge configuration model for your system.

When it is nearing update time, use

```
drush features-update-all
```

Relationships

I like you, do you like me?

Relationships

How do *we* do it?

User Relationships

complex relationships

drupal.
org/project/user_relationships

- Complex setup, Complex relations
- Overkill for simple relationships
- UI and UX is rough, at best.
- Mature (as in Old. Not Exportable!)

Flag Friend

Two-way approval

drupal.org/project/flag_friend

- Simple UI (uses flag.module, for the most part)
- Only supports basic friendship

Organic Groups

Moderated Membership

drupal.org/project/og

- Allows grouping of users and content
- Provides generic way to link all content to group (and give access to only the members of that group)

Relations

API Module. Needs Interface, but

- Provides a new Entity type: Relation. Relations are fieldable, but don't have to be.
- Relations can be "symmetrical" (Friends) or "directional" (Followers)

The Purpose of Relationships

- Define Content Privacy: Who can see my content?
  Who can I limit the visibility of my content to?

- Define Content Subscription: What do I see?
  Who's content shows up on my feed?

- Define Permissions: Who can do what with me?
  Write on my Wall, Direct Message me, Request Friendship, etc.

Why do you want relationships?

What value does adding relationships to your site have?

Without functionality behind it, relationships are pointless.
Be sure to think about what creating a relationship does.

Is this going to help or hurt?

Without an extremely polished and fluid User Experience, managing another social list will become a chore.

Types of Relationships

flag.module or user_relationships.module

Following

- One way. Follow a person (or thing) to subscribe to their content. They don't have to follow you back.
- Does not necessarily make sense to use for access control.

flag_friend.module or user_relationships.module

Friendship

- Both users approve of the relationship.
- Usually a misnomer, because its the only relationship on a site.
- Difficult to maintain for popular users (celebrities, etc.)
- Usually comes with a way to make content "friends only". This implies there is content that is public. (or at least user_access("access content")

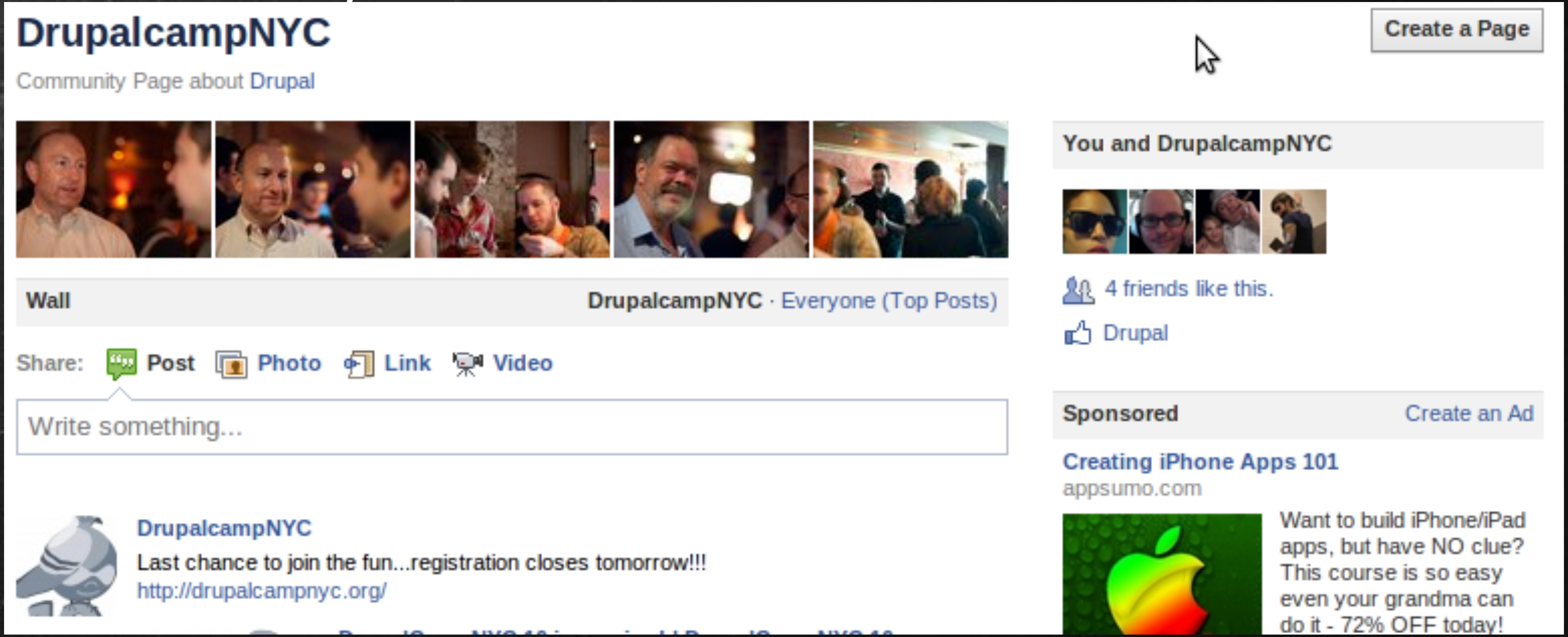
relation.module or user_relationships.module

- Both users are subscribed to each others content.

Types of Relationships

Automatic & Discovered

- Users can be related by their mutual interests, location, or any other piece of data you collect.
- Use this to help users discover one another, even if they are already friends.



DrupalcampNYC

Community Page about Drupal

Wall

DrupalcampNYC · Everyone (Top Posts)

Share: Post Photo Link Video

Write something...

DrupalcampNYC

Last chance to join the fun...registration closes tomorrow!!!

http://drupalcampnyc.org/

Create a Page

You and DrupalcampNYC

4 friends like this.

Drupal

Sponsored Create an Ad

Creating iPhone Apps 101

appsumo.com

Want to build iPhone/iPad apps, but have NO clue? This course is so easy even your grandma can do it - 72% OFF today!

Communication

"You've Got Mail"
"Can you hear me now?"
"Facebook Me"

Communication

How do *we* do it?

Private Messages

another inbox

drupal.org/project/privatemsg

- A simple messaging system.
- UI and UX is rough. Really rough.
- Mature (as in Old.)

Notifications

Don't forget to remind me.

drupal.org/project/notifications
drupal.org/project/messaging

- Manage your email templates with care.
- There may be more than one person to email when the time comes.
- Don't spam your users. Allow personalization and use sensible def

User Profiles

user.module is ok with fields, profile types are better.

upal.org/project/profile2

- Drupal 7 Core allows fields on a User.
- This gets lumped in with all of the account settings a Drupal user already has to deal with.
- Profile2.module allows "Profile Types" to be created, just like node types, allowing one user to have one of each type of

Communication

How do ***we*** do it?

Status Posts & Post to "Wall"

Targeted but open messages.

Custom Node Type, User Reference Field, & Code

- Allow friends to post something targeted at one person, but visible to many.
- Starts discussions among groups of friends

Sharing

Take this node and share it.

Custom Node Type, Node Reference Field, & Code

- The ability to reference and share an existing piece of content on your site

privatemsg.module

another inbox

Private Messages is a very old and reliable module that looks and acts its age.

You will want to heavily alter the presentation and display logic.

Think long and hard about whether you really need to give your users another inbox to check.

You will never make it as smooth

"Status & Wall Posts"

node.type: status

Fields

title: 255 Characters max in the database. You don't need a field for a tweet.

uid: The author of this post.

field_ref_user: The target of this post.

- If posted on another user's "wall", field_ref_user will = that user.
- defaults to node:uid for easier filtering: a profile page uses this field for an argument
- **This user needs ability to delete this post (and comments!)**
- **This user's friends should be able to see this post (possibly)**

Usage

node/add/status? no. (unless you want a popup)

Use hook_block_info() and hook_block_view() to create a block. Load drupal_get_form('status_node_form') into the block. Place the block in the content region on (almost) every page. Voila! Status Form.

hook_form_alter() allows you to do lots of other modifications to the form to

"Share Posts"

node.type: share

Fields

title: 255 Characters max in the database. You don't need a field for a tweet.

uid: The author of this post.

field_ref_user: Same as a status node.

field_ref_node: A node reference field storing the node to be shared.

- Drupal paths ignore additional arguments: node/add/share/12354/self returns the same page as node/add/share
- Create a link on every share-able node type to "node/add/share/$NID/self"
- Use form_alter() to set and hide this field, just like field_ref_user.

Its nice to add the option to "share with a friend" in addition to "post to my wall". if (arg(4) != self), UNHIDE field_ref_user and use the autocomplete for "enter a friend
to share with"!

User Profiles

Drupal 7 Goodness

Fields in Core!

Fields possible on all Entities: Nodes, Users, Terms, Files, etc.

Taxonomy is linked to nodes via "Term Reference" fields

You can add "Term Reference" fields to Users, giving us the ability to link users to content and each other through similar terms.

Add image and link fields to Terms.

Profile2.module

System for creating "Profile Types", allowing grouping of fields for each user. Emulates the old profile.module which had "categories" of user profiles.

User Profiles

Drupal 7 Goodness

Taxonomy as "Things to Like"

Using taxonomy terms as the storage for "things people like" opens up a lot of possibilities.

We are able to build lists of the "Most popular things".

We are able to show you content that is tagged with those things. (written on that thing's "Wall")

We are able to match you with other users based on those things.

Distribution

Feed Me!

Distribution

How do *we* do it?

Search API

Index Everything.

drupal.org/project/search_api

- So much more than search...
- Basically an interface to No-SQL data sources (Solr, Mongo, you name it)
- Can index all entities
- All Views can be Search API powered

Entity API

Specifically...

hook_entity_property_info()

drupal.org/project/entity

- Everything Drupal 7 wanted to be
- Define properties with arbitrary getters and setters
- Automatically access all properties in Search API
- References Fields connects the referenced entities and can index attached fields.

Apache Solr

Document-based index storage = fast

upal.
g/project/search_api_solr

- Not MySQL means better, stronger, faster
- Powered by Views means Easy, Powerful, Flexible
- Search API means

Distribution Channels

How are your users fed content?

Your Feed

A list of content and activity created by users you are related to or want to see content and activity from.

May or may not include posts on friends walls by non-friends.

Your "Wall"

Your posts (when not targeted at another user), and other users posts on your wall.

Access control is important here. Users must be able to delete posts on their "walls".

Additional Feeds

Usually, having at least one other thing to let users find content with is helpful.

Groups, Lists, Taxonomy Terms...

Something.

Entity API

Enhance the Entities

The Entity API allows you to do many things, including set extra properties of entities as they are saved.

In order to build things like a "Friends Feed", a "Following Feed" or a "Wall", we can build extra properties of a user and a node.

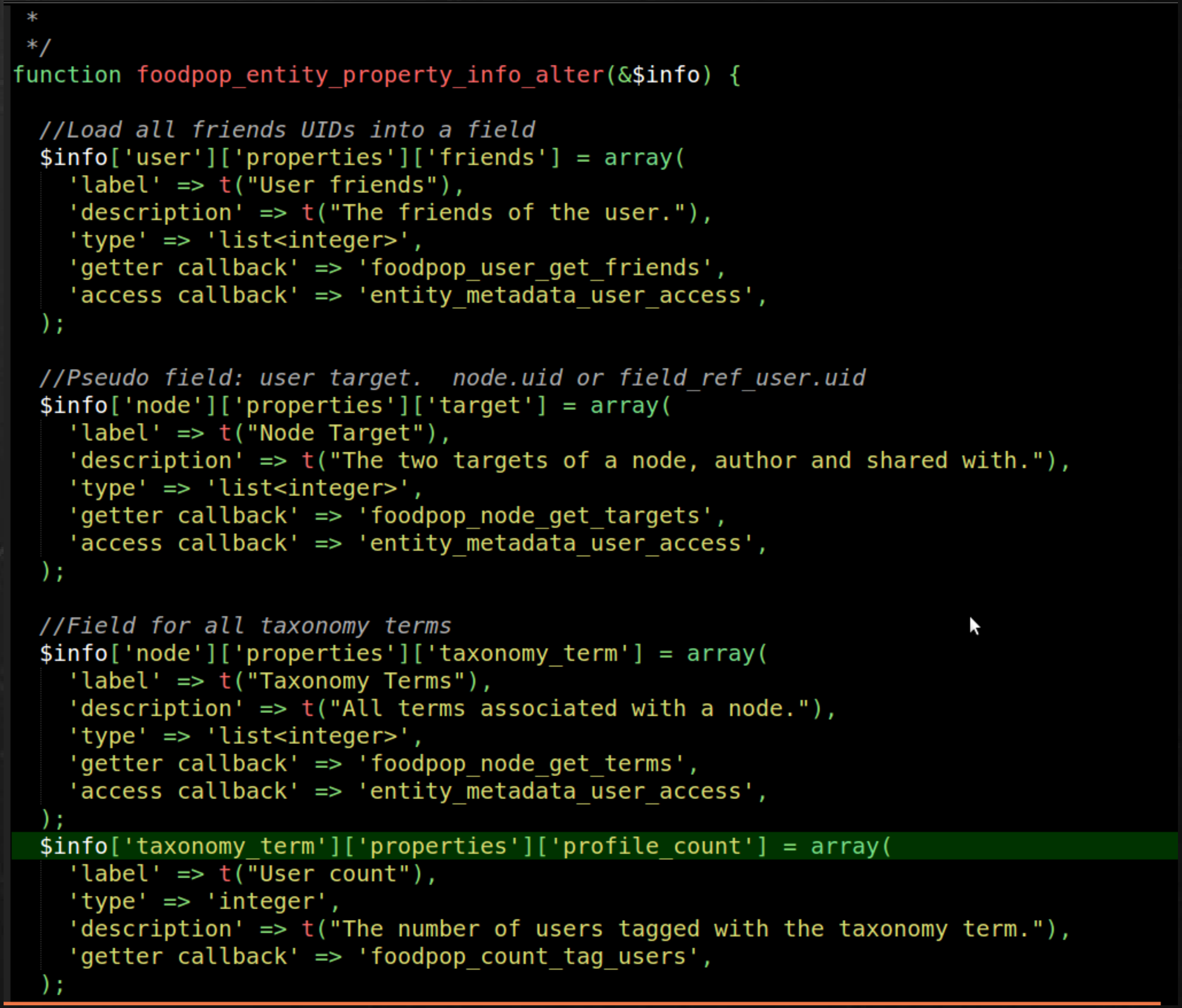
Those properties are then loaded by the Search API into its index.

Entity API

hook_entity_property_info_alter()

By creating the "friends" property of the user entity, we will be able to load it when we index a node, which means each node has a list of "friends" it should be shown to in their friends feed.

Then, we can build a Search API View with the current $user->uid as the



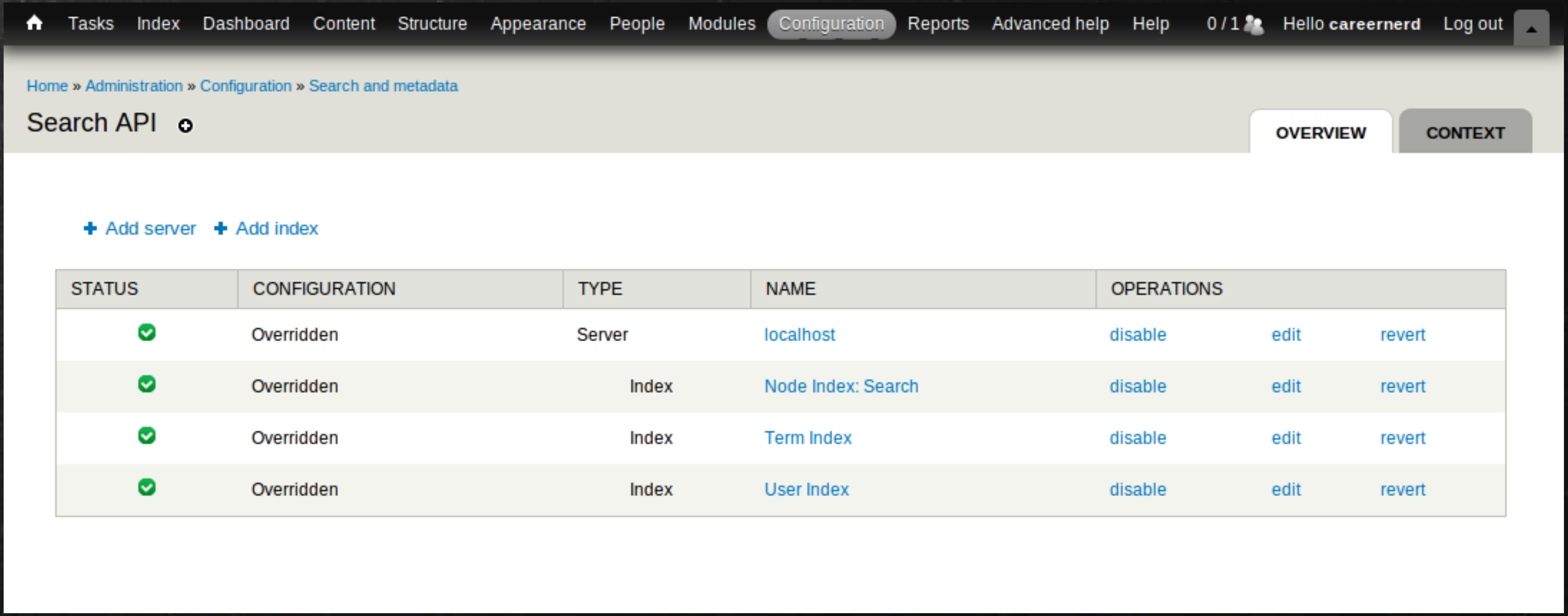
```
 *
 */
function foodpop_entity_property_info_alter(&$info) {

  //Load all friends UIDs into a field
  $info['user']['properties']['friends'] = array(
    'label' => t("User friends"),
    'description' => t("The friends of the user."),
    'type' => 'list<integer>',
    'getter callback' => 'foodpop_user_get_friends',
    'access callback' => 'entity_metadata_user_access',
  );

  //Pseudo field: user target.  node.uid or field_ref_user.uid
  $info['node']['properties']['target'] = array(
    'label' => t("Node Target"),
    'description' => t("The two targets of a node, author and shared with."),
    'type' => 'list<integer>',
    'getter callback' => 'foodpop_node_get_targets',
    'access callback' => 'entity_metadata_user_access',
  );

  //Field for all taxonomy terms
  $info['node']['properties']['taxonomy_term'] = array(
    'label' => t("Taxonomy Terms"),
    'description' => t("All terms associated with a node."),
    'type' => 'list<integer>',
    'getter callback' => 'foodpop_node_get_terms',
    'access callback' => 'entity_metadata_user_access',
  );
  $info['taxonomy_term']['properties']['profile_count'] = array(
    'label' => t("User count"),
    'type' => 'integer',
    'description' => t("The number of users tagged with the taxonomy term."),
    'getter callback' => 'foodpop_count_tag_users',
  );
```

Search API

Extrapolate the Index

Creates multiple Indexes and allows multiple servers

Have an index for each thing you want to search or display from a Search API backend like Solr.



STATUS	CONFIGURATION	TYPE	NAME	OPERATIONS		
✓	Overridden	Server	localhost	disable	edit	revert
✓	Overridden	Index	Node Index: Search	disable	edit	revert
✓	Overridden	Index	Term Index	disable	edit	revert
✓	Overridden	Index	User Index	disable	edit	revert

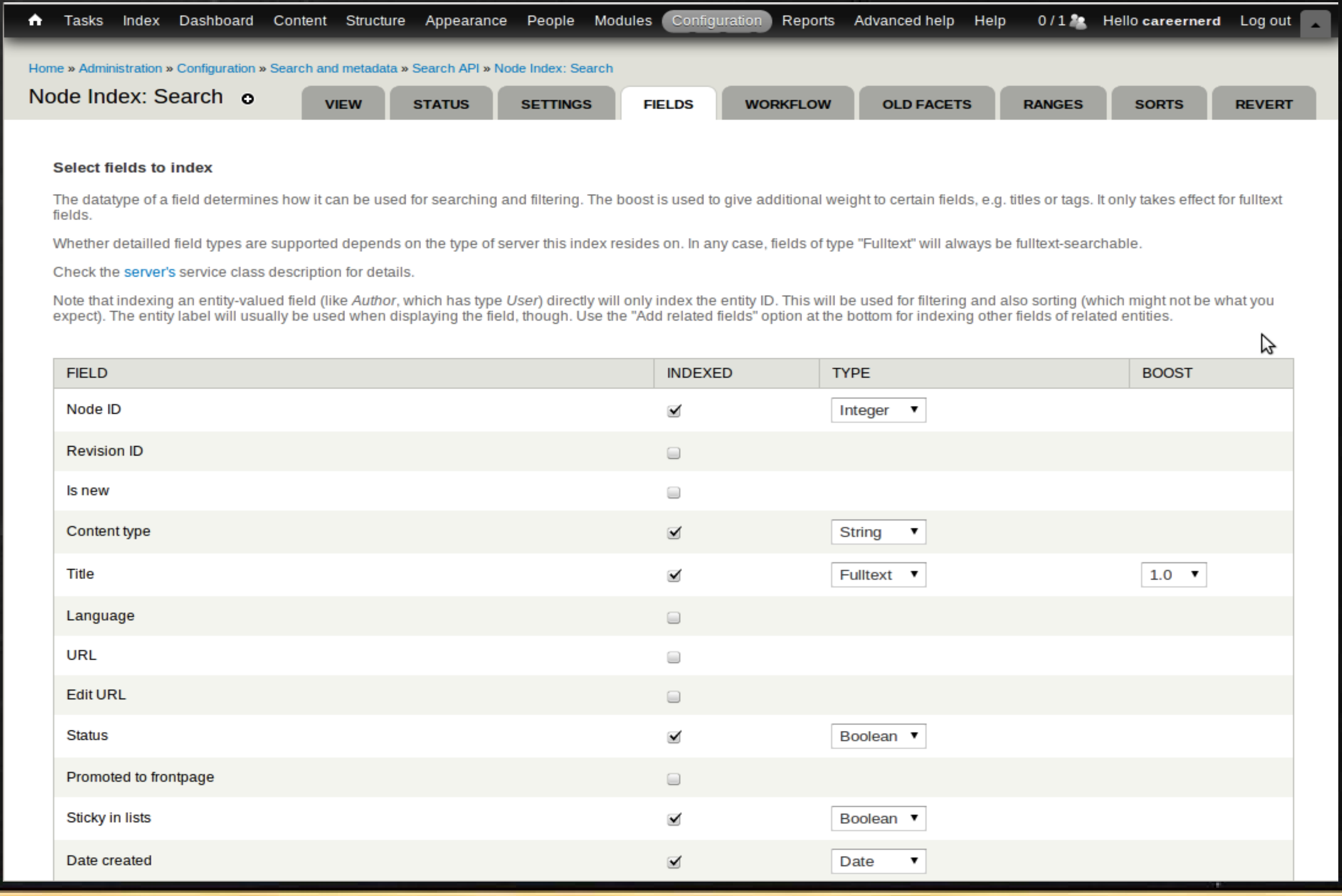
Search API

Extrapolate the Index

Choose your entity type, then choose your fields. Entity relationships allow you to branch out and load related fields as well.

Allows you to build specific indexes with only the fields you need.

The indexes and Fields you create then become available in Views for your building pleasure.



Home » Administration » Configuration » Search and metadata » Search API » Node Index: Search

Node Index: Search

Select fields to index

The datatype of a field determines how it can be used for searching and filtering. The boost is used to give additional weight to certain fields, e.g. titles or tags. It only takes effect for fulltext fields.

Whether detailled field types are supported depends on the type of server this index resides on. In any case, fields of type "Fulltext" will always be fulltext-searchable.

Check the server's service class description for details.

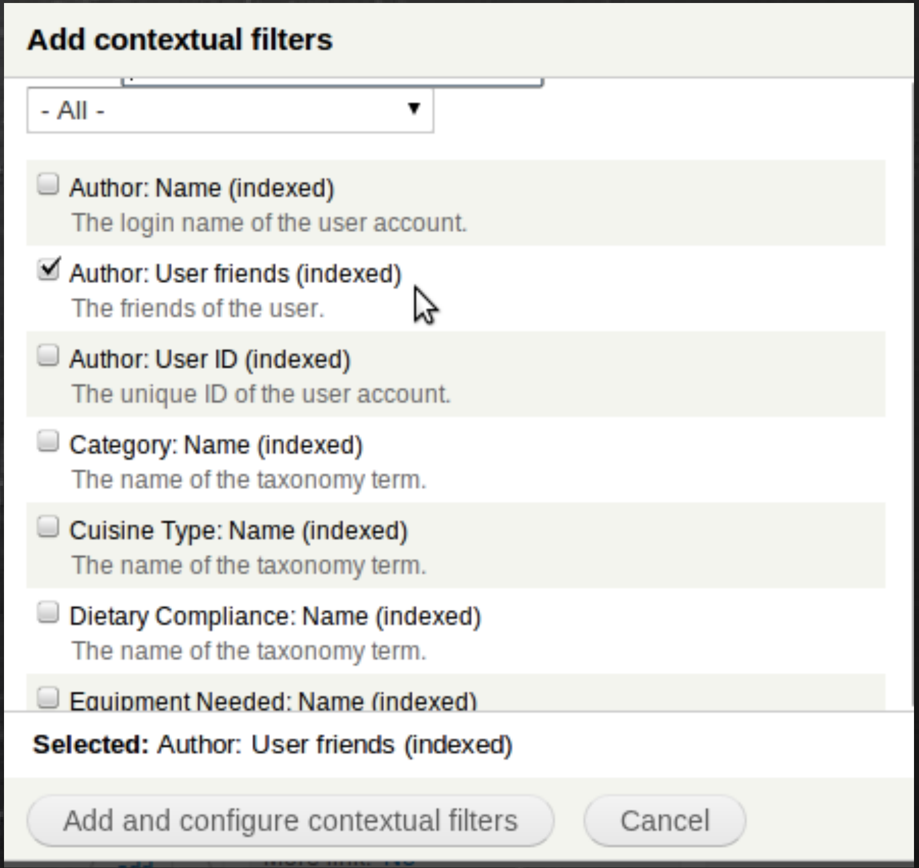
Note that indexing an entity-valued field (like *Author*, which has type *User*) directly will only index the entity ID. This will be used for filtering and also sorting (which might not be what you expect). The entity label will usually be used when displaying the field, though. Use the "Add related fields" option at the bottom for indexing other fields of related entities.

FIELD	INDEXED	TYPE	BOOST
Node ID	✓	Integer	
Revision ID	☐		
Is new	☐		
Content type	✓	String	
Title	✓	Fulltext	1.0
Language	☐		
URL	☐		
Edit URL	☐		
Status	✓	Boolean	
Promoted to frontpage	☐		
Sticky in lists	✓	Boolean	
Date created	✓	Date	

"Friend Feed"

Content from any of your friends

A view with the a Contextual filter for User Friends, that defaults to $global->user;

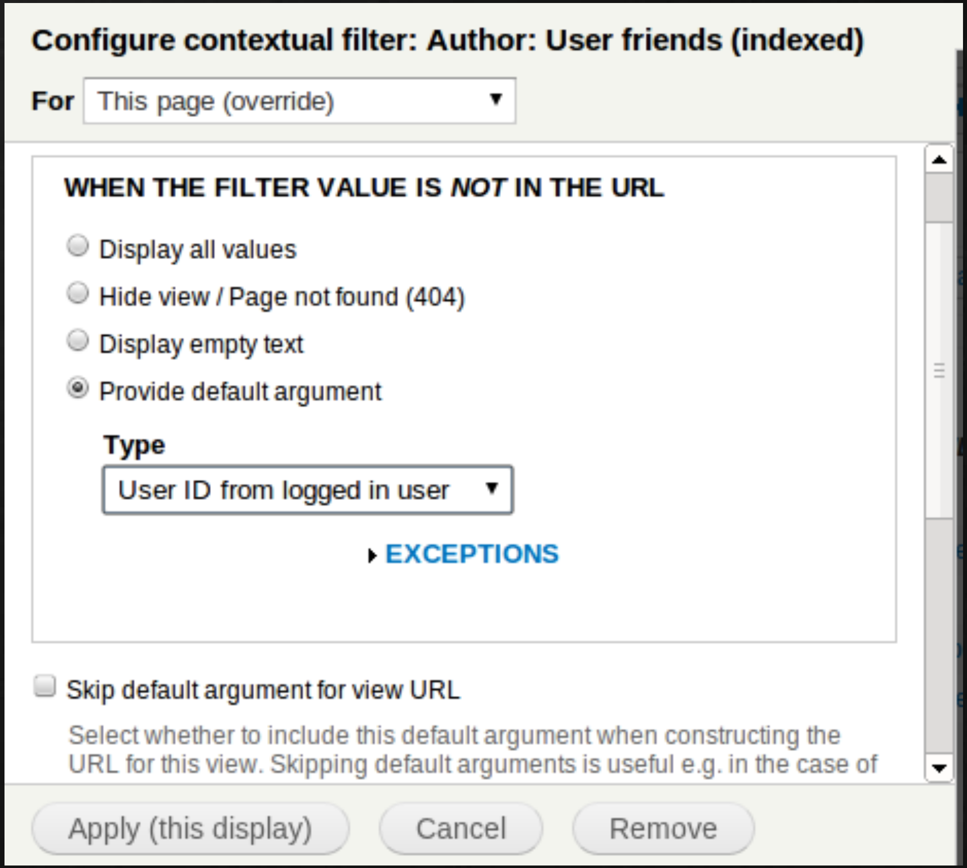


Add contextual filters

- All -

- [ ] Author: Name (indexed)
  The login name of the user account.
- [✓] Author: User friends (indexed)
  The friends of the user.
- [ ] Author: User ID (indexed)
  The unique ID of the user account.
- [ ] Category: Name (indexed)
  The name of the taxonomy term.
- [ ] Cuisine Type: Name (indexed)
  The name of the taxonomy term.
- [ ] Dietary Compliance: Name (indexed)
  The name of the taxonomy term.
- [ ] Equipment Needed: Name (indexed)

Selected: Author: User friends (indexed)



Configure contextual filter: Author: User friends (indexed)

For This page (override)

WHEN THE FILTER VALUE IS *NOT* IN THE URL

- ○ Display all values
- ○ Hide view / Page not found (404)
- ○ Display empty text
- ● Provide default argument

Type
User ID from logged in user

▸ EXCEPTIONS

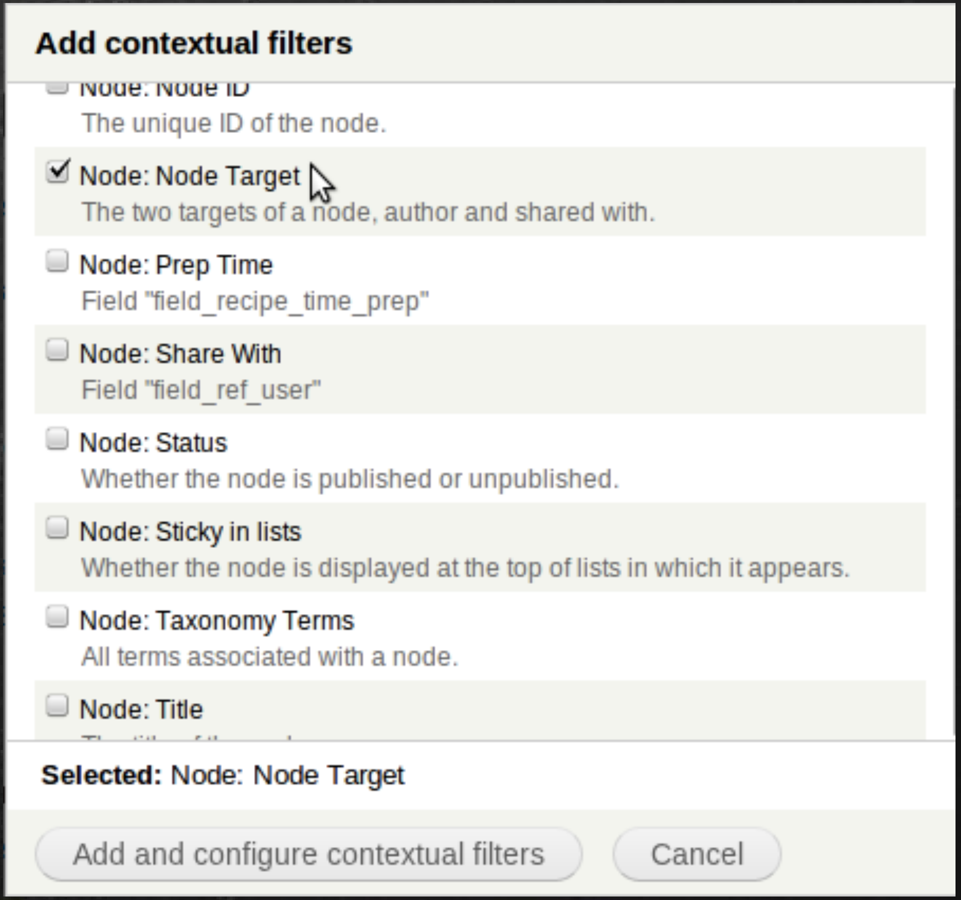
- [ ] Skip default argument for view URL
  Select whether to include this default argument when constructing the URL for this view. Skipping default arguments is useful e.g. in the case of

"Wall Feed"

Your Posts (not on friends walls) and Friends Posts (on your wall)

A view with the a Contextual filter for Node: Target, that defaults to the User ID in the current URL.

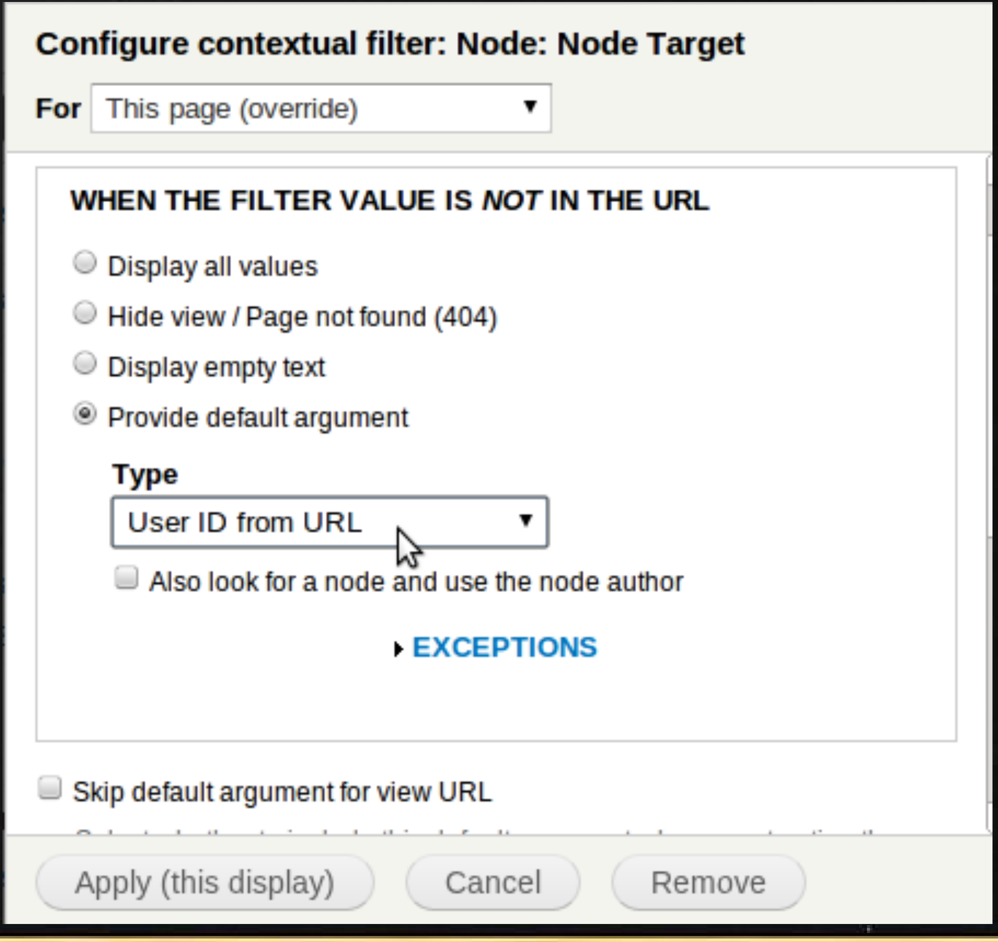
Path: user/%   Overrides existing "User Profile Page"



Add contextual filters

- [ ] Node: Node ID
  The unique ID of the node.
- [x] Node: Node Target
  The two targets of a node, author and shared with.
- [ ] Node: Prep Time
  Field "field_recipe_time_prep"
- [ ] Node: Share With
  Field "field_ref_user"
- [ ] Node: Status
  Whether the node is published or unpublished.
- [ ] Node: Sticky in lists
  Whether the node is displayed at the top of lists in which it appears.
- [ ] Node: Taxonomy Terms
  All terms associated with a node.
- [ ] Node: Title

Selected: Node: Node Target



Configure contextual filter: Node: Node Target

For This page (override)

WHEN THE FILTER VALUE IS *NOT* IN THE URL

- ○ Display all values
- ○ Hide view / Page not found (404)
- ○ Display empty text
- ● Provide default argument

Type
User ID from URL

- [ ] Also look for a node and use the node author

- [ ] Skip default argument for view URL

Privacy & Permissions

"Control! Control! You must have control!"

Why do you need Privacy & Permissions Control?

How much control do your users *really* need?

A balance must be struck between security and access. With privacy and permissions control, micromanagement can also become a problem. Make it as easy as possible for your users.

Remember: Every Site is Different!

Privacy & Permissions

How do ***we*** do it?

Node Access

Low Level Protection.

Core System. Requires contrib modules to take advantage.

Safely hide content using modules.
Nodes only.
System-Wide access control.

User Relationships or Flag Friend Access

Relationship modules provide access control.

Plugs into node_access. Provides user interface for choosing what relationships can view your nodes.

Profile Fields can be used for user settings.

Or just use your custom module, a form_alter and

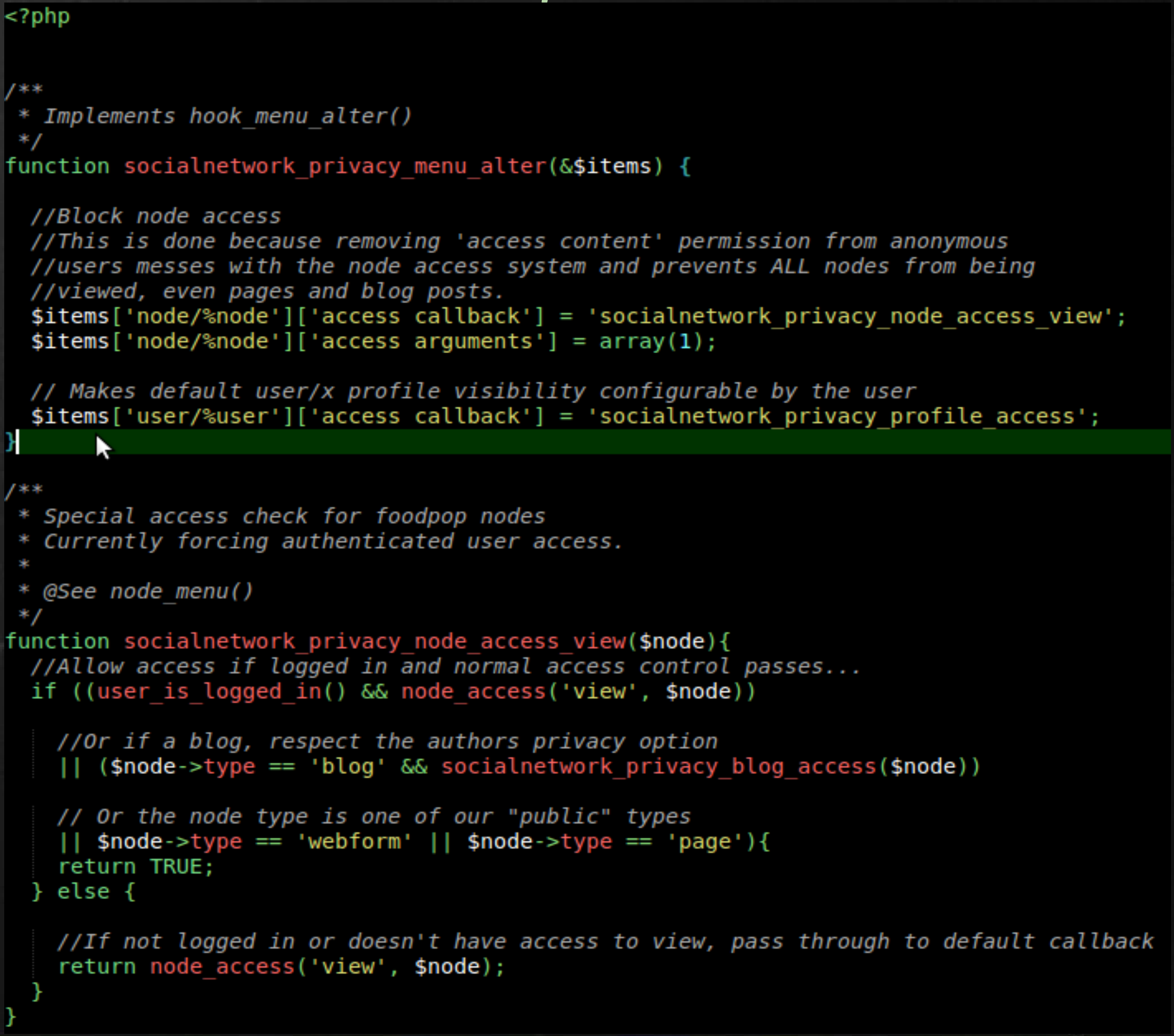
Create settings based on how your site works and what is best for your users.

Privacy & Permissions

Common Access Checkpoints

hook_menu_alter()

Add your own function to "access callback" to check if your users are allowed to view the page on some special conditions.



```php
<?php


/**
 * Implements hook_menu_alter()
 */
function socialnetwork_privacy_menu_alter(&$items) {

  //Block node access
  //This is done because removing 'access content' permission from anonymous
  //users messes with the node access system and prevents ALL nodes from being
  //viewed, even pages and blog posts.
  $items['node/%node']['access callback'] = 'socialnetwork_privacy_node_access_view';
  $items['node/%node']['access arguments'] = array(1);

  // Makes default user/x profile visibility configurable by the user
  $items['user/%user']['access callback'] = 'socialnetwork_privacy_profile_access';
}

/**
 * Special access check for foodpop nodes
 * Currently forcing authenticated user access.
 *
 * @See node_menu()
 */
function socialnetwork_privacy_node_access_view($node){
  //Allow access if logged in and normal access control passes...
  if ((user_is_logged_in() && node_access('view', $node))

    //Or if a blog, respect the authors privacy option
    || ($node->type == 'blog' && socialnetwork_privacy_blog_access($node))

    // Or the node type is one of our "public" types
    || $node->type == 'webform' || $node->type == 'page'){
    return TRUE;
  } else {

    //If not logged in or doesn't have access to view, pass through to default callback
    return node_access('view', $node);
  }
}
```

Privacy & Permissions

Common Access Checkpoints

Build Elements

(a.k.a. Forms API a.k.a "render-able arrays")

To hide an element in a form or a build array:

```
$element['#access'] = FALSE;
```

hook_node_access()

Simple True/False access check.

hook_menu_alter()

```
$item['user/%user']['access callback'] = 'custom_access_check';
```

Privacy & Permissions

Saving User Settings

User or Profile2 Fields

More configurable. (Can be good or bad!)
Data is more accessible. Uses Field API storage.
Don't forget to "Manage Display" and hide setting fields!
$user->field_setting[LANGUAGE_NONE][0]['value']

$user->data

Not accessible with views or queries at all.
Can only load if you've loaded the user.
Data is serialized into the {users}.data table column.

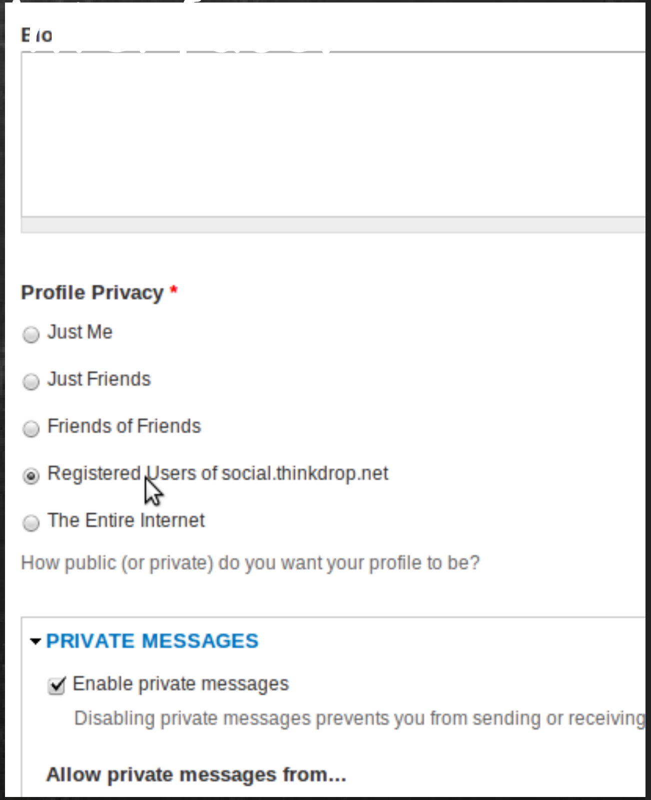
Custom Module

ok_schema() + hook_user_load() + hook_user_insert() + hook_user_update()

Privacy & Permissions

Saving User Settings

Using a Field offers flexibility and a nice friendly, familiar,



E /o

Profile Privacy *

- ○ Just Me
- ○ Just Friends
- ○ Friends of Friends
- ● Registered Users of social.thinkdrop.net
- ○ The Entire Internet

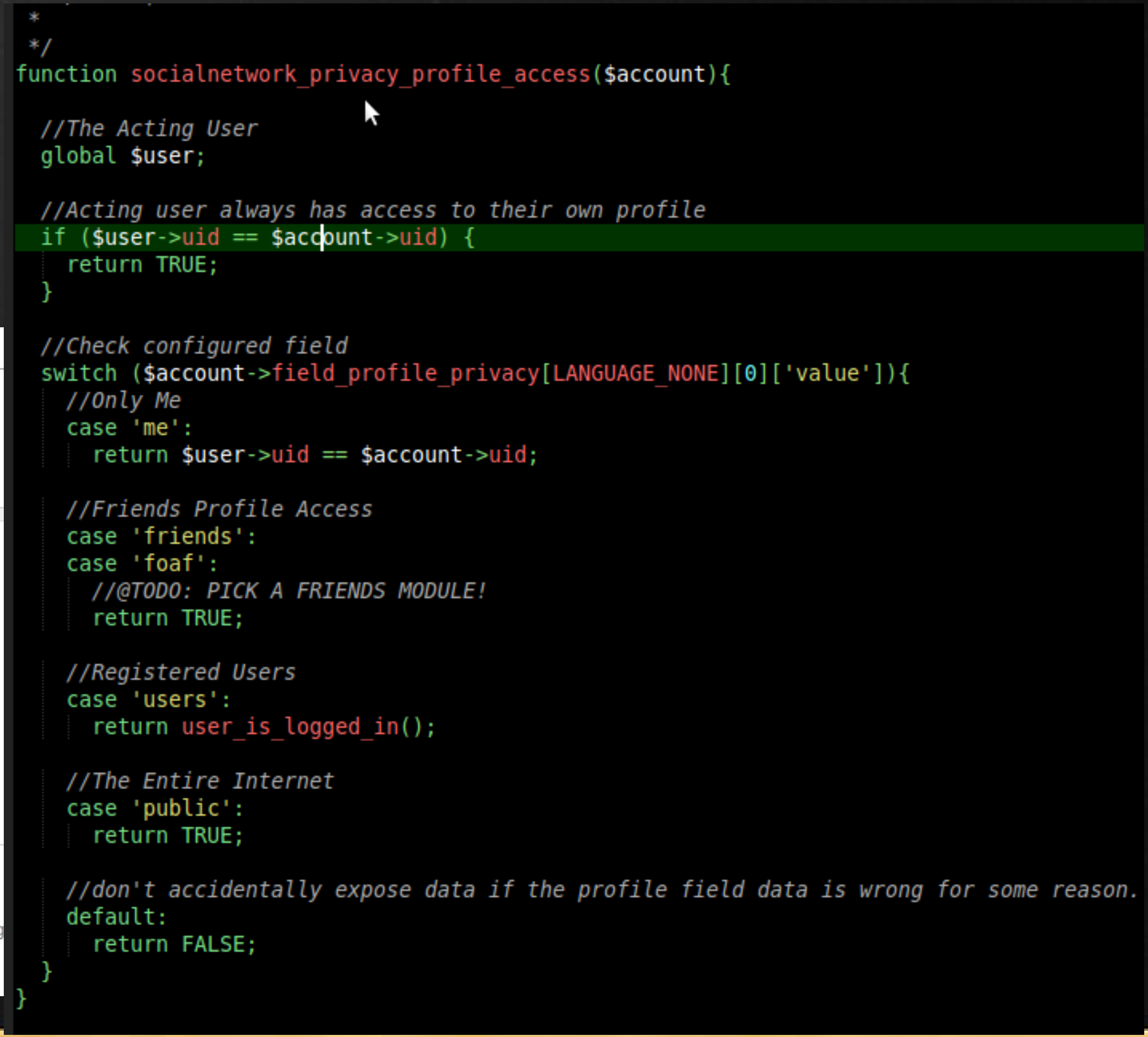
How public (or private) do you want your profile to be?

▾ **PRIVATE MESSAGES**

☑ Enable private messages

Disabling private messages prevents you from sending or receiving

Allow private messages from...



```
 *
 */
function socialnetwork_privacy_profile_access($account){

  //The Acting User
  global $user;

  //Acting user always has access to their own profile
  if ($user->uid == $account->uid) {
    return TRUE;
  }

  //Check configured field
  switch ($account->field_profile_privacy[LANGUAGE_NONE][0]['value']){
    //Only Me
    case 'me':
      return $user->uid == $account->uid;

    //Friends Profile Access
    case 'friends':
    case 'foaf':
      //@TODO: PICK A FRIENDS MODULE!
      return TRUE;

    //Registered Users
    case 'users':
      return user_is_logged_in();

    //The Entire Internet
    case 'public':
      return TRUE;

    //don't accidentally expose data if the profile field data is wrong for some reason.
    default:
      return FALSE;
  }
}
```

Activity

Nodes are boring.
What are you doing?

Activity

How do *we* do it?

Activity.module

Not Recommended. Can't get it to work on Drupal 7.
drupal.org/project/activity

Heartbeat.module

*Better, but still more trouble than
its worth.*
drupal.org/project/heartbeat

Statuses.module

Formerly Facebook-Style Statuses drupal.org/project/statuses

We don't recommend any of the existing Drupal Contrib "activity" type modules. They are buggy, and very rigid because of their custom storage and code. Drupal FieldAPI can be leveraged to build a new system.

A node can represent an activity, and there are a number of reasons this is ideal.

Activity

How do ***we*** do it?

With Nodes!

Myth: Node's are "Heavy".

Reality: Node's have a lot of features that other entities (or custom "objects" like a "heartbeat" message) don't have, that are required for certain functionality.

- node.uid: Node's have an owner, who is granted higher permissions over the node.
- **Node Access:** Nodes have access control, so you can keep activity private using other node access modules.
- **Comments:** Nodes can be commented on.
- **They are Nodes.** This means they can be easily listed in a View with other Nodes, so you don't have

Activity

node.type: activity

Fields

title: Not used. Display is processed with code.

uid: The actor of the activity

field_activity_type: A machine-name defining what type it is. The message theming is changed based on this.

field_activity_ref_nodes & field_activity_ref_terms & field_activity_ref_users: References to each type of object that might be connected to an activity.

Use situational logic to look up

Activity Types:

comment
profile changes
flag_favorite
flag_follow
flag_like
friends
photos
signup
videos

Integration

Don't build a walled garden.

Integration

Janrain RPX

Multiple Network Integration

drupal.org/project/rpx

Easiest "Social Network Registration & Login" service we have used.

Built and maintained by the Janrain company

Twitter

It just works

Imports tweets, allows users to tweet when they create nodes.

drupal.org/project/twitter

Simple module, not the best UI, but has decent Tweet storage, views support, and it just works.

Services

Build your own API.

drupal.org/project/services

Pluggable API system.

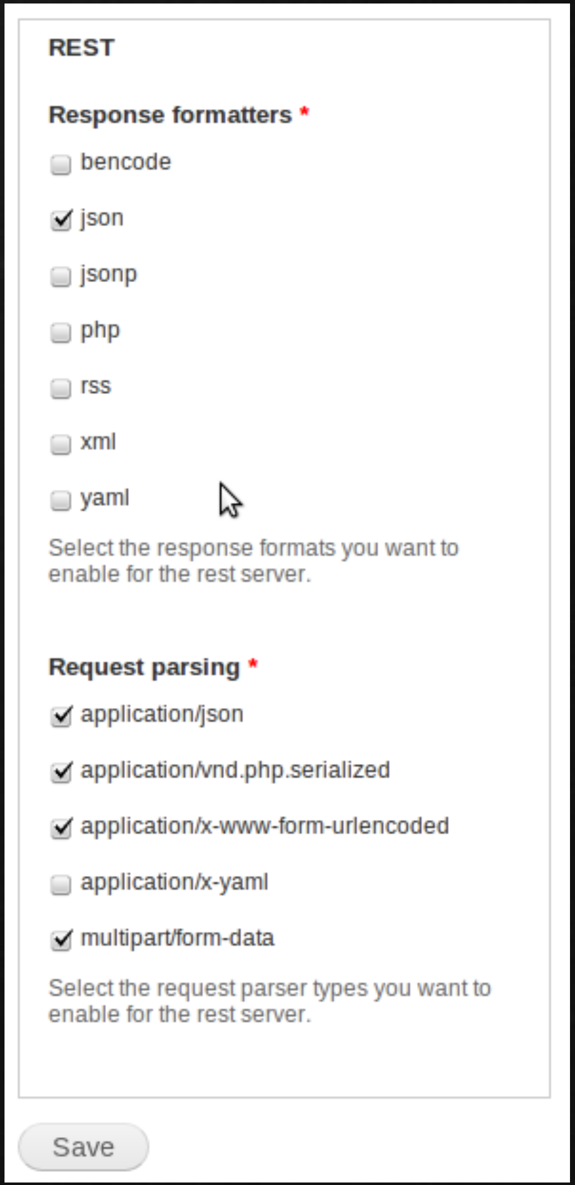
REST/JSON/XML/Whatever

Opens up all basic Drupal functions like user registration, login, update...

Services API

Framework for Web Services

- Pluggable system for Web Service Endpoints
- Allows 3rd Party App developers to interact with your web app.
- Lets you provide multiple response formats, authentication types, and more while abstracting the actual API commands.
- Most commands simply pass through to forms their respective forms, allowing altering via hook_form_alter()


REST

Response formatters *

- [ ] bencode
- [x] json
- [ ] jsonp
- [ ] php
- [ ] rss
- [ ] xml
- [ ] yaml

Select the response formats you want to enable for the rest server.

Request parsing *

- [x] application/json
- [x] application/vnd.php.serialized
- [x] application/x-www-form-urlencoded
- [ ] application/x-yaml
- [x] multipart/form-data

Select the request parser types you want to enable for the rest server.

Questions?

Building Social Networks

NYC Camp

July 21 2012

Jonathan Pugh
jon@thinkdrop.net
twitter.com/careernerd
Jon Pugh on drupal.org

THINKDROP
http://thinkdrop.net
twitter.com/thinkdropNYC
facebook.com/thinkdrop

http://thinkdrop.net/socialnetworks
to access these slides and for more information.

WE'RE HIRING
http://thinkdrop.net/apply