Directory: intelora-core/intelora.sh

```
#!/usr/bin/env bash

SOURCE="${BASH_SOURCE[0]}"
while [ -h "$SOURCE" ]; do
 DIR="$( cd -P "$( dirname "$SOURCE" )" && pwd )"
 SOURCE="$(readlink "$SOURCE")"
 [[ $SOURCE != /* ]] && SOURCE="$DIR/$SOURCE"
done
DIR="$( cd -P "$( dirname "$SOURCE" )" && pwd )"
SCRIPTS="$DIR/scripts"

function usage {
 echo
 echo "Quickly start, stop or restart Intelora's esential
services in detached screens"
 echo
 echo "usage: $0 [-h] (start [-v|-c]|stop|restart)"
 echo "     -h          this help message"
 echo "     start       starts intelora-service, intelora-skills,
intelora-voice and intelora-cli in quiet mode"
 echo "     start -v    starts intelora-service, intelora-skills
and intelora-voice"
 echo "     start -c    starts intelora-service, intelora-skills
and intelora-cli"
 echo "     stop        stops intelora-service, intelora-skills
and intelora-voice"
 echo "     restart     restarts intelora-service, intelora-skills
and intelora-voice"
 echo
 echo "screen tips:"
 echo "          run 'screen -list' to see all running screens"
 echo "          run 'screen -r <screen-name>' (e.g. 'screen -r
intelora-service') to reatach a screen"
 echo "          press ctrl + a, ctrl + d to detace the screen
again"
 echo "          See the screen man page for more details"
 echo
}

#mkdir -p $SCRIPTS/logs
mkdir -p $DIR/logs

function verify-start {
  if ! screen -list | grep -q "$1";
  then
    echo "$1 failed to start. The log is below:"
    echo
    tail $DIR/logs/$1.log
  exit 1
  fi
}

function start-intelora {
 screen -mdS intelora-$1$2 -c $SCRIPTS/intelora-$1.screen
$DIR/scripts/start.sh $1 $2
 sleep 1
 verify-start intelora-$1$2
 echo "Intelora $1$2 started"
}

function stop-intelora {
  if screen -list | grep -q "$1";
  then
    screen -XS intelora-$1 quit
    echo "Intelora $1 stopped"
  fi
}

function restart-intelora {
  if screen -list | grep -q "quiet";
  then
    $0 stop
    sleep 1
    $0 start
  elif screen -list | grep -q "cli" && ! screen -list | grep -q
"quiet";
  then
    $0 stop
    sleep 1
    $0 start -c
  elif screen -list | grep -q "voice" && ! screen -list | grep -q
"quiet";
  then
    $0 stop
    sleep 1
    $0 start -v
  else
    echo "An error occurred"
  fi
}

set -e

if [[ -z "$1" || "$1" == "-h" ]]
then
 usage
 exit 1
elif [[ "$1" == "start" && -z "$2" ]]
then
 start-intelora service
 start-intelora skills
 start-intelora voice
 start-intelora cli --quiet
 exit 0
elif [[ "$1" == "start" && "$2" == "-v" ]]
then
 start-intelora service
 start-intelora skills
 start-intelora voice
 exit 0
elif [[ "$1" == "start" && "$2" == "-c" ]]
then
 start-intelora service
 start-intelora skills
 start-intelora cli
 exit 0
elif [[ "$1" == "stop" && -z "$2" ]]
then
 stop-intelora service
 stop-intelora skills
 stop-intelora voice
 stop-intelora cli
 exit 0
elif [[ "$1" == "restart" && -z "$2" ]]
then
 restart-intelora
 exit 0
else
 usage
 exit 1
fi
```

Directory: intelora-core/build/build_host_setup_arch.sh

```
#!/usr/bin/env bash

sudo pacman -S \
  git \
  python2 \
  python2-pip \
  python2-setuptools \
  python2-virtualenv \
  python2-gobject \
  python-virtualenvwrapper \
  libtool \
  libffi \
  openssl \
  autoconf \
  bison \
  swig \
  glib2 \
  s3cmd \
  portaudio \
  mpg123 \
  screen \
  flac \
  curl

# upgrade virtualenv to latest from pypi
sudo pip2 install --upgrade virtualenv
```

Directory: intelora-core/build/build_host_setup_debian.sh

```
#!/usr/bin/env bash

sudo apt-get install -y \
  git \
  python \
  python-dev \
  python-setuptools \
  python-virtualenv \
  python-gobject-dev \
  virtualenvwrapper \
  libtool \
  libffi-dev \
  libssl-dev \
  autoconf \
  bison \
  swig \
  libglib2.0-dev \
  s3cmd \
  portaudio19-dev \
  mpg123 \
  screen \
  flac \
  curl
```

Directory: intelora-core/build/build_host_setup_fedora.sh

```
#!/usr/bin/env bash

sudo rpm -Uvh
http://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-23.noarch.rpm

sudo dnf install -y \
  git \
  python \
  python-devel \
  python-pip \
  python-setuptools \
  python-virtualenv \
  pygobject2-devel \
  python-virtualenvwrapper \
  libtool \
  libffi-devel \
  openssl-devel \
  autoconf \
  bison \
  swig \
  glib2-devel \
  s3cmd \
  portaudio-devel \
  mpg123 \
  screen \
  curl

# upgrade virtualenv to latest from pypi
sudo pip install --upgrade virtualenv
```

Directory: intelora-core/doc/generate_sdk_docs.py

```
import os
import pdoc

__author__ = 'seanfitz'

DOCS_NAME = "mycroft-skills-sdk"

DOC_OUTPUT_DIR = "build/doc/%s/html" % DOCS_NAME

documented_sdk_modules = [
  "mycroft.configuration",
  "mycroft.dialog",
  "mycroft.filesystem",
  "mycroft.session",
  "mycroft.util",
  "mycroft.util.log"
]


def module_to_docpath(module_name):
  module_source_dir = module_name.replace(".", "/")
  module_doc_dir = os.path.join(DOC_OUTPUT_DIR,
module_source_dir)
  base_module_name =
os.path.basename(module_doc_dir)
  if not os.path.isdir(module_source_dir):
    d = os.path.dirname(module_doc_dir)
    try:
      os.makedirs(d)
    except OSError:
      pass
    return os.path.join(d, base_module_name + '.m.html')
```

```
    else:
        try:
            os.makedirs(module_doc_dir)
        except OSError:
            pass
        return os.path.join(module_doc_dir, 'index.html')


def main():
    for m in documented_sdk_modules:
        html = pdoc.html(m, allsubmodules=True)
        with open(module_to_docpath(m), 'w') as f:
            f.write(html)
    import mycroft
    mycroft.__all__ = [m[8:] for m in
documented_sdk_modules]
    root_module = pdoc.Module(mycroft)
    html = root_module.html(external_links=False,
link_prefix='', source=True)
    with open(module_to_docpath("mycroft"), 'w') as f:
        f.write(html)


if __name__ == "__main__":
    main()
```

Directory: intelora-core/logs/intelora-cli.log

```
Starting cli --quiet
2016-12-24 11:38:29,119 - mycroft.configuration - DEBUG -
Configuration
'/home/paul/Irene/mycroft/configuration/mycroft.conf'
loaded
2016-12-24 11:38:29,126 - mycroft.configuration - DEBUG -
Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:38:29,127 - mycroft.configuration - DEBUG -
Configuration '/home/paul/.mycroft/mycroft.conf' not
found
2016-12-24 11:38:29,467 -
requests.packages.urllib3.connectionpool - INFO - Starting
new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:38:30,818 -
requests.packages.urllib3.connectionpool - DEBUG - "GET
/v1/device/c14b1936-4458-4859-9cab-
ebb56da21994/setting HTTP/1.1" 200 2887
 Carnegie Mellon University, Copyright (c) 1999-2011, all
rights reserved
 version: mimic-2.0.0-release Dec 2014 (http://cmuflite.org)
Input:
2016-12-24 11:38:31,055 - mycroft.messagebus.client.ws -
INFO - Connected
what time s
Input:
^C2016-12-24 11:38:46,711 - CLIClient - ERROR -
Traceback (most recent call last):
 File "/home/paul/Irene/mycroft/client/text/main.py", line
61, in main
   line = sys.stdin.readline()
KeyboardInterrupt
Traceback (most recent call last):
 File "/home/paul/Irene/mycroft/client/text/main.py", line
72, in <module>
   main()
 File "/home/paul/Irene/mycroft/client/text/main.py", line
67, in main
   event_thread.exit()
AttributeError: 'Thread' object has no attribute 'exit'
Starting cli --quiet
2016-12-24 11:40:18,081 - mycroft.configuration - DEBUG -
Configuration
'/home/paul/Irene/mycroft/configuration/mycroft.conf'
loaded
2016-12-24 11:40:18,082 - mycroft.configuration - DEBUG -
Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:40:18,082 - mycroft.configuration - DEBUG -
Configuration '/home/paul/.mycroft/mycroft.conf' not
found
2016-12-24 11:40:18,405 -
requests.packages.urllib3.connectionpool - INFO - Starting
new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:40:19,728 -
requests.packages.urllib3.connectionpool - DEBUG - "GET
/v1/device/c14b1936-4458-4859-9cab-
ebb56da21994/setting HTTP/1.1" 200 2887
 Carnegie Mellon University, Copyright (c) 1999-2011, all
rights reserved
 version: mimic-2.0.0-release Dec 2014 (http://cmuflite.org)
Input:
2016-12-24 11:40:20,016 - mycroft.messagebus.client.ws -
INFO - Connected
what imt e       tell me about my chemical romance
Input:
^C2016-12-24 11:40:47,760 - CLIClient - ERROR -
Traceback (most recent call last):
 File "/home/paul/Irene/mycroft/client/text/main.py", line
61, in main
   line = sys.stdin.readline()
KeyboardInterrupt
Traceback (most recent call last):
 File "/home/paul/Irene/mycroft/client/text/main.py", line
72, in <module>
   main()
 File "/home/paul/Irene/mycroft/client/text/main.py", line
67, in main
   event_thread.exit()
AttributeError: 'Thread' object has no attribute 'exit'
```

Directory: intelora-core/logs/intelora-service.log

```
Starting service
2016-12-24 11:38:26,417 -
requests.packages.urllib3.connectionpool - INFO - Starting
new HTTPS connection (1): api.mycroft.ai
/home/paul/Irene/init/start.sh: line 26: 28965 Terminated
PYTHONPATH=${TOP} python ${SCRIPT} $@
Starting service
2016-12-24 11:40:15,424 -
requests.packages.urllib3.connectionpool - INFO - Starting
new HTTPS connection (1): api.mycroft.ai
/home/paul/Irene/init/start.sh: line 26: 29461 Terminated
PYTHONPATH=${TOP} python ${SCRIPT} $@
```

Directory: intelora-core/logs/intelora-skills.log

```
Starting skills
2016-12-24 11:38:26,908 - mycroft.configuration - DEBUG -
Configuration
```

'/home/paul/Irene/mycroft/configuration/mycroft.conf' loaded
2016-12-24 11:38:26,909 - mycroft.configuration - DEBUG - Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:38:26,910 - mycroft.configuration - DEBUG - Configuration '/home/paul/.mycroft/mycroft.conf' not found
2016-12-24 11:38:27,204 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:38:28,495 - requests.packages.urllib3.connectionpool - DEBUG - "GET /v1/device/c14b1936-4458-4859-9cab-ebb56da21994/setting HTTP/1.1" 200 2887
2016-12-24 11:38:28,697 - mycroft.messagebus.client.ws - INFO - Connected

** (process:28980): [1;33mWARNING[0m **: Trying to register gtype 'GMountMountFlags' as enum when in fact it is of type 'GFlags'

** (process:28980): [1;33mWARNING[0m **: Trying to register gtype 'GDriveStartFlags' as enum when in fact it is of type 'GFlags'

** (process:28980): [1;33mWARNING[0m **: Trying to register gtype 'GSocketMsgFlags' as enum when in fact it is of type 'GFlags'
2016-12-24 11:38:31,963 - Skills - DEBUG - {"type": "connected", "data": {}, "context": null}
2016-12-24 11:38:31,965 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timer", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:38:31,971 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timers", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:38:31,979 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarm", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:38:31,985 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarms", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:38:31,987 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:38:31,991 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all my", "end": "AlarmSkillAmount", "alias_of": "all"}, "context": null}
2016-12-24 11:38:31,993 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:38:31,997 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "AlarmSkillAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:38:32,003 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:38:32,006 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "AlarmSkillAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:38:32,011 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:38:32,014 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:38:32,016 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "off", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,023 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,025 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,034 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "erase", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:32,040 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:32,047 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "delete", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:32,055 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remove", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:32,061 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timer", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:38:32,067 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timers", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:38:32,076 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarm", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:38:32,082 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarms", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:38:32,093 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "show", "end": "AlarmSkillListVerb"}, "context": null}
2016-12-24 11:38:32,104 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "list", "end": "AlarmSkillListVerb"}, "context": null}
2016-12-24 11:38:32,111 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<AlarmSkillAmount>\\d+)"}, "context": null}
2016-12-24 11:38:32,141 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillCreateVerb", "AlarmSkillCreateVerb"]], "optional": [], "name": "AlarmSkillCreateIntent"}, "context": null}
2016-12-24 11:38:32,152 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillListVerb", "AlarmSkillListVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [["AlarmSkillAmount", "AlarmSkillAmount"]], "name": "AlarmSkillListIntent"}, "context": null}
2016-12-24 11:38:32,155 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillDeleteVerb", "AlarmSkillDeleteVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [["AlarmSkillAmount", "AlarmSkillAmount"]], "name": "AlarmSkillDeleteIntent"}, "context": null}
2016-12-24 11:38:32,159 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillStopVerb", "AlarmSkillStopVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [], "name": "AlarmSkillStopIntent"}, "context": null}

2016-12-24 11:38:32,163 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "record", "end": "AudioRecordSkillKeyword"}, "context": null}
2016-12-24 11:38:32,165 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "recording", "end": "AudioRecordSkillKeyword"}, "context": null}
2016-12-24 11:38:32,168 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,175 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,180 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:38:32,198 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,201 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "play", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,208 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "replay", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,214 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "playback", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,222 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reproduce", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,228 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "running", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,231 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "playing", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,235 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "replaying", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,238 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reproducing", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:38:32,243 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillIntent"}, "context": null}
2016-12-24 11:38:32,247 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillStopVerb", "AudioRecordSkillStopVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillStopIntent"}, "context": null}
2016-12-24 11:38:32,254 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillPlayVerb", "AudioRecordSkillPlayVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillPlayIntent"}, "context": null}
2016-12-24 11:38:32,260 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillStopVerb", "AudioRecordSkillStopVerb"], ["AudioRecordSkillPlayVerb", "AudioRecordSkillPlayVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillStopPlayIntent"}, "context": null}
2016-12-24 11:38:32,267 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "config", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:38:32,273 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "configuration", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:38:32,278 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "setting", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:38:32,286 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "update", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:38:32,292 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reload", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:38:32,301 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sync", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:38:32,307 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "synchronize", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:38:32,314 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ConfigurationSkillKeyword", "ConfigurationSkillKeyword"], ["ConfigurationSkillUpdateVerb", "ConfigurationSkillUpdateVerb"]], "optional": [], "name": "UpdateConfigurationIntent"}, "context": null}
2016-12-24 11:38:32,319 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what time is it", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,323 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what is the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,327 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what's the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,347 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "whats the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,357 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what time is", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,363 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "time is it", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:38:32,367 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what's the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:38:32,373 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "whats the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:38:32,381 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what day is it", "end": "DateKeyword"}, "context": null}
2016-12-24 11:38:32,387 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what is the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:38:32,391 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:38:32,403 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["TimeKeyword", "TimeKeyword"]], "optional": [["Location", "Location"]], "name": "TimeIntent"}, "context": null}

2016-12-24 11:38:32,406 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "SearchKeyword"}, "context": null}
2016-12-24 11:38:32,410 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "find", "end": "SearchKeyword"}, "context": null}
2016-12-24 11:38:32,413 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "google", "end": "Website"}, "context": null}
2016-12-24 11:38:32,419 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "facebook", "end": "Website"}, "context": null}
2016-12-24 11:38:32,425 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "amazon", "end": "Website"}, "context": null}
2016-12-24 11:38:32,430 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "youtube", "end": "Website"}, "context": null}
2016-12-24 11:38:32,441 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "yahoo", "end": "Website"}, "context": null}
2016-12-24 11:38:32,446 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wikipedia", "end": "Website"}, "context": null}
2016-12-24 11:38:32,451 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "ebay", "end": "Website"}, "context": null}
2016-12-24 11:38:32,459 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "twitter", "end": "Website"}, "context": null}
2016-12-24 11:38:32,464 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "go", "end": "Website"}, "context": null}
2016-12-24 11:38:32,470 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "craigslist", "end": "Website"}, "context": null}
2016-12-24 11:38:32,479 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reddit", "end": "Website"}, "context": null}
2016-12-24 11:38:32,484 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "linkedin", "end": "Website"}, "context": null}
2016-12-24 11:38:32,497 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "netflix", "end": "Website"}, "context": null}
2016-12-24 11:38:32,505 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "live", "end": "Website"}, "context": null}
2016-12-24 11:38:32,512 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bing", "end": "Website"}, "context": null}
2016-12-24 11:38:32,513 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pinterest", "end": "Website"}, "context": null}
2016-12-24 11:38:32,515 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "espn", "end": "Website"}, "context": null}
2016-12-24 11:38:32,518 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "imgur", "end": "Website"}, "context": null}
2016-12-24 11:38:32,520 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tumblr", "end": "Website"}, "context": null}
2016-12-24 11:38:32,523 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chase", "end": "Website"}, "context": null}
2016-12-24 11:38:32,536 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cnn", "end": "Website"}, "context": null}
2016-12-24 11:38:32,539 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "paypal", "end": "Website"}, "context": null}
2016-12-24 11:38:32,553 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "instagram", "end": "Website"}, "context": null}
2016-12-24 11:38:32,563 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "blogspot", "end": "Website"}, "context": null}
2016-12-24 11:38:32,572 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "apple", "end": "Website"}, "context": null}
2016-12-24 11:38:32,583 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "launch", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:38:32,591 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "open", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:38:32,593 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:38:32,599 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "for (?P<SearchTerms>.*)"}, "context": null}
2016-12-24 11:38:32,604 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "for (?P<SearchTerms>.*) on"}, "context": null}
2016-12-24 11:38:32,614 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<SearchTerms>.*) on"}, "context": null}
2016-12-24 11:38:32,624 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cd/dvd creator", "end": "Application"}, "context": null}
2016-12-24 11:38:32,627 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cd", "end": "Application"}, "context": null}
2016-12-24 11:38:32,639 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse & touch\u00adpad", "end": "Application"}, "context": null}
2016-12-24 11:38:32,644 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse", "end": "Application"}, "context": null}
2016-12-24 11:38:32,647 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bleachbit", "end": "Application"}, "context": null}
2016-12-24 11:38:32,653 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "assistive technologies", "end": "Application"}, "context": null}
2016-12-24 11:38:32,657 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "assistive", "end": "Application"}, "context": null}
2016-12-24 11:38:32,660 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "glade", "end": "Application"}, "context": null}
2016-12-24 11:38:32,666 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome system monitor", "end": "Application"}, "context": null}
2016-12-24 11:38:32,670 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:38:32,674 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "help", "end": "Application"}, "context": null}

2016-12-24 11:38:32,684 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop search", "end": "Application"}, "context": null}
2016-12-24 11:38:32,688 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:38:32,695 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cheese", "end": "Application"}, "context": null}
2016-12-24 11:38:32,703 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "customize look and feel", "end": "Application"}, "context": null}
2016-12-24 11:38:32,708 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "customize", "end": "Application"}, "context": null}
2016-12-24 11:38:32,726 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "feh", "end": "Application"}, "context": null}
2016-12-24 11:38:32,730 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "robots", "end": "Application"}, "context": null}
2016-12-24 11:38:32,735 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "home folder", "end": "Application"}, "context": null}
2016-12-24 11:38:32,737 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "home", "end": "Application"}, "context": null}
2016-12-24 11:38:32,741 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "brasero", "end": "Application"}, "context": null}
2016-12-24 11:38:32,745 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gparted", "end": "Application"}, "context": null}
2016-12-24 11:38:32,748 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,752 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font", "end": "Application"}, "context": null}
2016-12-24 11:38:32,756 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pri\u00adva\u00adcy", "end": "Application"}, "context": null}
2016-12-24 11:38:32,761 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mines", "end": "Application"}, "context": null}
2016-12-24 11:38:32,766 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome shell extension preferences", "end": "Application"}, "context": null}
2016-12-24 11:38:32,771 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:38:32,779 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "image viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,781 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "image", "end": "Application"}, "context": null}
2016-12-24 11:38:32,784 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mutter", "end": "Application"}, "context": null}
2016-12-24 11:38:32,787 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:38:32,790 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thunderbird", "end": "Application"}, "context": null}
2016-12-24 11:38:32,795 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about mate", "end": "Application"}, "context": null}
2016-12-24 11:38:32,798 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about", "end": "Application"}, "context": null}
2016-12-24 11:38:32,802 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firefox", "end": "Application"}, "context": null}
2016-12-24 11:38:32,806 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}
2016-12-24 11:38:32,810 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice writer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,814 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:32,818 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "prin\u00adters", "end": "Application"}, "context": null}
2016-12-24 11:38:32,821 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth manager", "end": "Application"}, "context": null}
2016-12-24 11:38:32,824 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:38:32,826 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "co\u00adlor", "end": "Application"}, "context": null}
2016-12-24 11:38:32,828 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:38:32,831 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "no\u00adti\u00adfi\u00adca\u00adtions", "end": "Application"}, "context": null}
2016-12-24 11:38:32,833 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mplayer media player", "end": "Application"}, "context": null}
2016-12-24 11:38:32,839 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mplayer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,841 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tweak tool", "end": "Application"}, "context": null}
2016-12-24 11:38:32,844 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tweak", "end": "Application"}, "context": null}
2016-12-24 11:38:32,848 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "users", "end": "Application"}, "context": null}
2016-12-24 11:38:32,851 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate dictionary", "end": "Application"}, "context": null}
2016-12-24 11:38:32,856 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:32,859 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "take screenshot", "end": "Application"}, "context": null}
2016-12-24 11:38:32,863 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "take", "end": "Application"}, "context": null}

2016-12-24 11:38:32,866 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake preferences", "end": "Application"}, "context": null}
2016-12-24 11:38:32,870 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake", "end": "Application"}, "context": null}
2016-12-24 11:38:32,873 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "file management", "end": "Application"}, "context": null}
2016-12-24 11:38:32,876 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "file", "end": "Application"}, "context": null}
2016-12-24 11:38:32,878 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hdspconf", "end": "Application"}, "context": null}
2016-12-24 11:38:32,884 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "polari", "end": "Application"}, "context": null}
2016-12-24 11:38:32,889 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice math", "end": "Application"}, "context": null}
2016-12-24 11:38:32,891 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:32,895 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice draw", "end": "Application"}, "context": null}
2016-12-24 11:38:32,898 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:32,908 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "characters", "end": "Application"}, "context": null}
2016-12-24 11:38:32,917 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "windows", "end": "Application"}, "context": null}
2016-12-24 11:38:32,924 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:38:32,928 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chess", "end": "Application"}, "context": null}
2016-12-24 11:38:32,931 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "web", "end": "Application"}, "context": null}
2016-12-24 11:38:32,934 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color profile viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,936 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:38:32,940 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about me", "end": "Application"}, "context": null}
2016-12-24 11:38:32,943 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about", "end": "Application"}, "context": null}
2016-12-24 11:38:32,946 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice xslt based filters", "end": "Application"}, "context": null}
2016-12-24 11:38:32,951 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:32,955 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "autorun prompt", "end": "Application"}, "context": null}
2016-12-24 11:38:32,957 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "autorun", "end": "Application"}, "context": null}
2016-12-24 11:38:32,959 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "passwords and keys", "end": "Application"}, "context": null}
2016-12-24 11:38:32,961 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "passwords", "end": "Application"}, "context": null}
2016-12-24 11:38:32,963 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnu image manipulation program", "end": "Application"}, "context": null}
2016-12-24 11:38:32,965 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnu", "end": "Application"}, "context": null}
2016-12-24 11:38:32,970 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "klotski", "end": "Application"}, "context": null}
2016-12-24 11:38:32,974 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pictures folder", "end": "Application"}, "context": null}
2016-12-24 11:38:32,978 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pictures", "end": "Application"}, "context": null}
2016-12-24 11:38:32,981 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dictionary", "end": "Application"}, "context": null}
2016-12-24 11:38:32,984 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mahjongg", "end": "Application"}, "context": null}
2016-12-24 11:38:32,988 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "log file viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:32,991 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "log", "end": "Application"}, "context": null}
2016-12-24 11:38:32,995 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution calendar", "end": "Application"}, "context": null}
2016-12-24 11:38:32,998 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution", "end": "Application"}, "context": null}
2016-12-24 11:38:33,002 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sysprof", "end": "Application"}, "context": null}
2016-12-24 11:38:33,008 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "theme installer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,011 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "theme", "end": "Application"}, "context": null}
2016-12-24 11:38:33,013 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate color selection", "end": "Application"}, "context": null}
2016-12-24 11:38:33,015 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,027 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cmake", "end": "Application"}, "context": null}
2016-12-24 11:38:33,030 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dis\u00adplays", "end": "Application"}, "context": null}
2016-12-24 11:38:33,033 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard shortcuts", "end": "Application"}, "context": null}

2016-12-24 11:38:33,036 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:38:33,040 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "net\u00adwork", "end": "Application"}, "context": null}
2016-12-24 11:38:33,043 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cosmos", "end": "Application"}, "context": null}
2016-12-24 11:38:33,046 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "quadrapassel", "end": "Application"}, "context": null}
2016-12-24 11:38:33,049 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse", "end": "Application"}, "context": null}
2016-12-24 11:38:33,052 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "control center", "end": "Application"}, "context": null}
2016-12-24 11:38:33,056 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "control", "end": "Application"}, "context": null}
2016-12-24 11:38:33,059 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sha\u00adring", "end": "Application"}, "context": null}
2016-12-24 11:38:33,063 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "computer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,065 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "view file", "end": "Application"}, "context": null}
2016-12-24 11:38:33,070 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "view", "end": "Application"}, "context": null}
2016-12-24 11:38:33,073 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating gnome", "end": "Application"}, "context": null}
2016-12-24 11:38:33,077 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating", "end": "Application"}, "context": null}
2016-12-24 11:38:33,081 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network tools", "end": "Application"}, "context": null}
2016-12-24 11:38:33,085 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:38:33,091 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "eye of mate image viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,096 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "eye", "end": "Application"}, "context": null}
2016-12-24 11:38:33,098 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote desktop viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,100 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote", "end": "Application"}, "context": null}
2016-12-24 11:38:33,103 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "Application"}, "context": null}
2016-12-24 11:38:33,108 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "accerciser", "end": "Application"}, "context": null}
2016-12-24 11:38:33,111 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rhythmbox", "end": "Application"}, "context": null}
2016-12-24 11:38:33,117 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qt v4l2 test utility", "end": "Application"}, "context": null}
2016-12-24 11:38:33,120 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qt", "end": "Application"}, "context": null}
2016-12-24 11:38:33,123 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "maps", "end": "Application"}, "context": null}
2016-12-24 11:38:33,126 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power statistics", "end": "Application"}, "context": null}
2016-12-24 11:38:33,130 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power", "end": "Application"}, "context": null}
2016-12-24 11:38:33,132 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2048", "end": "Application"}, "context": null}
2016-12-24 11:38:33,136 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nitrogen", "end": "Application"}, "context": null}
2016-12-24 11:38:33,139 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "panel", "end": "Application"}, "context": null}
2016-12-24 11:38:33,153 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "iagno", "end": "Application"}, "context": null}
2016-12-24 11:38:33,155 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "messenger for desktop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,158 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "messenger", "end": "Application"}, "context": null}
2016-12-24 11:38:33,161 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "archive manager", "end": "Application"}, "context": null}
2016-12-24 11:38:33,164 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "archive", "end": "Application"}, "context": null}
2016-12-24 11:38:33,167 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "root terminal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,170 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "root", "end": "Application"}, "context": null}
2016-12-24 11:38:33,174 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome mplayer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,177 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:38:33,180 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color picker", "end": "Application"}, "context": null}
2016-12-24 11:38:33,182 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:38:33,186 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pulseaudio volume control", "end": "Application"}, "context": null}
2016-12-24 11:38:33,188 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pulseaudio", "end": "Application"}, "context": null}
2016-12-24 11:38:33,191 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi vnc server browser", "end": "Application"}, "context": null}

2016-12-24 11:38:33,193 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:38:33,195 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firewall configuration", "end": "Application"}, "context": null}
2016-12-24 11:38:33,198 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firewall", "end": "Application"}, "context": null}
2016-12-24 11:38:33,200 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pop art squares", "end": "Application"}, "context": null}
2016-12-24 11:38:33,202 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,203 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hdspmixer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,205 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "screensaver", "end": "Application"}, "context": null}
2016-12-24 11:38:33,206 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "settings", "end": "Application"}, "context": null}
2016-12-24 11:38:33,208 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "electron", "end": "Application"}, "context": null}
2016-12-24 11:38:33,210 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm community edition", "end": "Application"}, "context": null}
2016-12-24 11:38:33,212 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm", "end": "Application"}, "context": null}
2016-12-24 11:38:33,215 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hitori", "end": "Application"}, "context": null}
2016-12-24 11:38:33,220 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thunderbird", "end": "Application"}, "context": null}
2016-12-24 11:38:33,237 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,242 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating", "end": "Application"}, "context": null}
2016-12-24 11:38:33,248 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qbittorrent", "end": "Application"}, "context": null}
2016-12-24 11:38:33,252 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop icons", "end": "Application"}, "context": null}
2016-12-24 11:38:33,256 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,260 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run software", "end": "Application"}, "context": null}
2016-12-24 11:38:33,264 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "Application"}, "context": null}
2016-12-24 11:38:33,267 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "aisleriot solitaire", "end": "Application"}, "context": null}
2016-12-24 11:38:33,272 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "aisleriot", "end": "Application"}, "context": null}
2016-12-24 11:38:33,276 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "logs", "end": "Application"}, "context": null}
2016-12-24 11:38:33,280 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "swell foop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,286 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "swell", "end": "Application"}, "context": null}
2016-12-24 11:38:33,300 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "taquin", "end": "Application"}, "context": null}
2016-12-24 11:38:33,302 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disks", "end": "Application"}, "context": null}
2016-12-24 11:38:33,304 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "engrampa archive manager", "end": "Application"}, "context": null}
2016-12-24 11:38:33,305 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "engrampa", "end": "Application"}, "context": null}
2016-12-24 11:38:33,307 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "i3", "end": "Application"}, "context": null}
2016-12-24 11:38:33,309 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "calculator", "end": "Application"}, "context": null}
2016-12-24 11:38:33,316 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "privilege granting", "end": "Application"}, "context": null}
2016-12-24 11:38:33,320 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "privilege", "end": "Application"}, "context": null}
2016-12-24 11:38:33,323 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tali", "end": "Application"}, "context": null}
2016-12-24 11:38:33,326 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "startup applications", "end": "Application"}, "context": null}
2016-12-24 11:38:33,328 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "startup", "end": "Application"}, "context": null}
2016-12-24 11:38:33,331 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "yaourt-gui", "end": "Application"}, "context": null}
2016-12-24 11:38:33,335 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "calendar", "end": "Application"}, "context": null}
2016-12-24 11:38:33,339 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "galculator", "end": "Application"}, "context": null}
2016-12-24 11:38:33,341 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:33,344 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nibbles", "end": "Application"}, "context": null}
2016-12-24 11:38:33,347 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk usage analyzer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,350 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:38:33,354 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "print preview", "end": "Application"}, "context": null}

2016-12-24 11:38:33,358 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "print", "end": "Application"}, "context": null}
2016-12-24 11:38:33,360 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth adapters", "end": "Application"}, "context": null}
2016-12-24 11:38:33,363 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:38:33,366 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system log", "end": "Application"}, "context": null}
2016-12-24 11:38:33,369 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system", "end": "Application"}, "context": null}
2016-12-24 11:38:33,372 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:38:33,375 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate disk usage analyzer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,378 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,381 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal file sharing", "end": "Application"}, "context": null}
2016-12-24 11:38:33,383 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,388 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "date & time", "end": "Application"}, "context": null}
2016-12-24 11:38:33,391 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "date", "end": "Application"}, "context": null}
2016-12-24 11:38:33,394 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "adobe flash player", "end": "Application"}, "context": null}
2016-12-24 11:38:33,399 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "adobe", "end": "Application"}, "context": null}
2016-12-24 11:38:33,409 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote desktop viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,412 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote", "end": "Application"}, "context": null}
2016-12-24 11:38:33,415 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice base", "end": "Application"}, "context": null}
2016-12-24 11:38:33,417 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:33,420 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "anjuta", "end": "Application"}, "context": null}
2016-12-24 11:38:33,423 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel preferences", "end": "Application"}, "context": null}
2016-12-24 11:38:33,426 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel", "end": "Application"}, "context": null}
2016-12-24 11:38:33,428 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gitg", "end": "Application"}, "context": null}
2016-12-24 11:38:33,431 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icc profile installer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,435 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icc", "end": "Application"}, "context": null}
2016-12-24 11:38:33,438 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pluma", "end": "Application"}, "context": null}
2016-12-24 11:38:33,440 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "vlc media player", "end": "Application"}, "context": null}
2016-12-24 11:38:33,443 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "vlc", "end": "Application"}, "context": null}
2016-12-24 11:38:33,449 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "volwheel", "end": "Application"}, "context": null}
2016-12-24 11:38:33,452 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "re\u00adgion & lan\u00adguage", "end": "Application"}, "context": null}
2016-12-24 11:38:33,456 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "re\u00adgion", "end": "Application"}, "context": null}
2016-12-24 11:38:33,460 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "Application"}, "context": null}
2016-12-24 11:38:33,464 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uxterm", "end": "Application"}, "context": null}
2016-12-24 11:38:33,468 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "help", "end": "Application"}, "context": null}
2016-12-24 11:38:33,470 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sudoku", "end": "Application"}, "context": null}
2016-12-24 11:38:33,474 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice calc", "end": "Application"}, "context": null}
2016-12-24 11:38:33,487 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:33,489 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "visual studio code", "end": "Application"}, "context": null}
2016-12-24 11:38:33,494 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "visual", "end": "Application"}, "context": null}
2016-12-24 11:38:33,499 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "character map", "end": "Application"}, "context": null}
2016-12-24 11:38:33,503 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "character", "end": "Application"}, "context": null}
2016-12-24 11:38:33,505 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "displays", "end": "Application"}, "context": null}
2016-12-24 11:38:33,508 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "five or more", "end": "Application"}, "context": null}
2016-12-24 11:38:33,511 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "five", "end": "Application"}, "context": null}
2016-12-24 11:38:33,514 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "de\u00adtails", "end": "Application"}, "context": null}

2016-12-24 11:38:33,517 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi zeroconf browser", "end": "Application"}, "context": null}
2016-12-24 11:38:33,519 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:38:33,525 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk image writer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,528 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:38:33,537 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tetravex", "end": "Application"}, "context": null}
2016-12-24 11:38:33,540 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "four-in-a-row", "end": "Application"}, "context": null}
2016-12-24 11:38:33,544 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system monitor", "end": "Application"}, "context": null}
2016-12-24 11:38:33,547 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system", "end": "Application"}, "context": null}
2016-12-24 11:38:33,550 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "documents", "end": "Application"}, "context": null}
2016-12-24 11:38:33,553 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "videos", "end": "Application"}, "context": null}
2016-12-24 11:38:33,556 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where am i?", "end": "Application"}, "context": null}
2016-12-24 11:38:33,559 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where", "end": "Application"}, "context": null}
2016-12-24 11:38:33,562 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "screenshot", "end": "Application"}, "context": null}
2016-12-24 11:38:33,565 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "clocks", "end": "Application"}, "context": null}
2016-12-24 11:38:33,567 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "books", "end": "Application"}, "context": null}
2016-12-24 11:38:33,570 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "music", "end": "Application"}, "context": null}
2016-12-24 11:38:33,573 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lights off", "end": "Application"}, "context": null}
2016-12-24 11:38:33,576 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lights", "end": "Application"}, "context": null}
2016-12-24 11:38:33,578 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network connections", "end": "Application"}, "context": null}
2016-12-24 11:38:33,581 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:38:33,587 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound recorder", "end": "Application"}, "context": null}
2016-12-24 11:38:33,590 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:38:33,592 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel", "end": "Application"}, "context": null}
2016-12-24 11:38:33,594 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard layout", "end": "Application"}, "context": null}
2016-12-24 11:38:33,596 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:38:33,599 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icon browser", "end": "Application"}, "context": null}
2016-12-24 11:38:33,602 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icon", "end": "Application"}, "context": null}
2016-12-24 11:38:33,605 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rhythmbox", "end": "Application"}, "context": null}
2016-12-24 11:38:33,610 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "text editor", "end": "Application"}, "context": null}
2016-12-24 11:38:33,613 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "text", "end": "Application"}, "context": null}
2016-12-24 11:38:33,617 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gtk+ demo", "end": "Application"}, "context": null}
2016-12-24 11:38:33,620 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gtk+", "end": "Application"}, "context": null}
2016-12-24 11:38:33,623 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "slack", "end": "Application"}, "context": null}
2016-12-24 11:38:33,626 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "manage printing", "end": "Application"}, "context": null}
2016-12-24 11:38:33,629 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "manage", "end": "Application"}, "context": null}
2016-12-24 11:38:33,637 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "terminal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,650 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate system monitor", "end": "Application"}, "context": null}
2016-12-24 11:38:33,653 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,656 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dconf editor", "end": "Application"}, "context": null}
2016-12-24 11:38:33,659 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dconf", "end": "Application"}, "context": null}
2016-12-24 11:38:33,661 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "document viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,665 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "document", "end": "Application"}, "context": null}
2016-12-24 11:38:33,670 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "popup notifications", "end": "Application"}, "context": null}
2016-12-24 11:38:33,673 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "popup", "end": "Application"}, "context": null}

2016-12-24 11:38:33,681 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi ssh server browser", "end": "Application"}, "context": null}
2016-12-24 11:38:33,683 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:38:33,686 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "back\u00adground", "end": "Application"}, "context": null}
2016-12-24 11:38:33,689 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "to do", "end": "Application"}, "context": null}
2016-12-24 11:38:33,693 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "to", "end": "Application"}, "context": null}
2016-12-24 11:38:33,696 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "inkscape", "end": "Application"}, "context": null}
2016-12-24 11:38:33,699 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "photos", "end": "Application"}, "context": null}
2016-12-24 11:38:33,702 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "on\u00adline accounts", "end": "Application"}, "context": null}
2016-12-24 11:38:33,705 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "on\u00adline", "end": "Application"}, "context": null}
2016-12-24 11:38:33,708 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power management", "end": "Application"}, "context": null}
2016-12-24 11:38:33,712 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power", "end": "Application"}, "context": null}
2016-12-24 11:38:33,715 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "po\u00adwer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,717 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "main menu", "end": "Application"}, "context": null}
2016-12-24 11:38:33,720 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "main", "end": "Application"}, "context": null}
2016-12-24 11:38:33,723 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "ipython", "end": "Application"}, "context": null}
2016-12-24 11:38:33,725 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm", "end": "Application"}, "context": null}
2016-12-24 11:38:33,727 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tixati", "end": "Application"}, "context": null}
2016-12-24 11:38:33,729 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network proxy", "end": "Application"}, "context": null}
2016-12-24 11:38:33,742 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:38:33,744 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uni\u00adver\u00adsal access", "end": "Application"}, "context": null}
2016-12-24 11:38:33,748 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uni\u00adver\u00adsal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,750 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake terminal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,753 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake", "end": "Application"}, "context": null}
2016-12-24 11:38:33,756 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database access control center", "end": "Application"}, "context": null}
2016-12-24 11:38:33,759 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database", "end": "Application"}, "context": null}
2016-12-24 11:38:33,762 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atomix", "end": "Application"}, "context": null}
2016-12-24 11:38:33,765 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:38:33,768 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,772 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font", "end": "Application"}, "context": null}
2016-12-24 11:38:33,776 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth transfer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,779 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:38:33,797 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "access prompt", "end": "Application"}, "context": null}
2016-12-24 11:38:33,802 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "access", "end": "Application"}, "context": null}
2016-12-24 11:38:33,805 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}
2016-12-24 11:38:33,808 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "terminator", "end": "Application"}, "context": null}
2016-12-24 11:38:33,811 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution", "end": "Application"}, "context": null}
2016-12-24 11:38:33,815 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "widget factory", "end": "Application"}, "context": null}
2016-12-24 11:38:33,818 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "widget", "end": "Application"}, "context": null}
2016-12-24 11:38:33,821 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network login", "end": "Application"}, "context": null}
2016-12-24 11:38:33,823 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:38:33,826 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice impress", "end": "Application"}, "context": null}
2016-12-24 11:38:33,829 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:38:33,832 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "key\u00adboard", "end": "Application"}, "context": null}
2016-12-24 11:38:33,835 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "appearance", "end": "Application"}, "context": null}

2016-12-24 11:38:33,839 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop sharing", "end": "Application"}, "context": null}
2016-12-24 11:38:33,842 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,846 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "htop", "end": "Application"}, "context": null}
2016-12-24 11:38:33,849 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search and indexing", "end": "Application"}, "context": null}
2016-12-24 11:38:33,852 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "Application"}, "context": null}
2016-12-24 11:38:33,856 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gitkraken", "end": "Application"}, "context": null}
2016-12-24 11:38:33,859 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database browser", "end": "Application"}, "context": null}
2016-12-24 11:38:33,863 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database", "end": "Application"}, "context": null}
2016-12-24 11:38:33,867 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "builder", "end": "Application"}, "context": null}
2016-12-24 11:38:33,870 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wa\u00adcom tab\u00adlet", "end": "Application"}, "context": null}
2016-12-24 11:38:33,872 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wa\u00adcom", "end": "Application"}, "context": null}
2016-12-24 11:38:33,874 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "devhelp", "end": "Application"}, "context": null}
2016-12-24 11:38:33,876 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk image mounter", "end": "Application"}, "context": null}
2016-12-24 11:38:33,879 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:38:33,881 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "files", "end": "Application"}, "context": null}
2016-12-24 11:38:33,886 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome shell", "end": "Application"}, "context": null}
2016-12-24 11:38:33,891 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:38:33,894 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate search tool", "end": "Application"}, "context": null}
2016-12-24 11:38:33,897 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,901 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal file sharing", "end": "Application"}, "context": null}
2016-12-24 11:38:33,910 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,918 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}
2016-12-24 11:38:33,920 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "preferred applications", "end": "Application"}, "context": null}
2016-12-24 11:38:33,924 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "preferred", "end": "Application"}, "context": null}
2016-12-24 11:38:33,931 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "contacts", "end": "Application"}, "context": null}
2016-12-24 11:38:33,935 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "blue\u00adtooth", "end": "Application"}, "context": null}
2016-12-24 11:38:33,938 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "empathy", "end": "Application"}, "context": null}
2016-12-24 11:38:33,941 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atril document viewer", "end": "Application"}, "context": null}
2016-12-24 11:38:33,944 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atril", "end": "Application"}, "context": null}
2016-12-24 11:38:33,946 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "xterm", "end": "Application"}, "context": null}
2016-12-24 11:38:33,949 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate terminal", "end": "Application"}, "context": null}
2016-12-24 11:38:33,952 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:38:33,959 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "marco", "end": "Application"}, "context": null}
2016-12-24 11:38:33,962 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spotify", "end": "Application"}, "context": null}
2016-12-24 11:38:33,969 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["LaunchKeyword", "LaunchKeyword"], ["Application", "Application"]], "optional": [], "name": "LaunchDesktopApplicationIntent"}, "context": null}
2016-12-24 11:38:33,972 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["LaunchKeyword", "LaunchKeyword"], ["Website", "Website"]], "optional": [], "name": "LaunchWebsiteIntent"}, "context": null}
2016-12-24 11:38:33,977 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SearchKeyword", "SearchKeyword"], ["Website", "Website"], ["SearchTerms", "SearchTerms"]], "optional": [], "name": "SearchWebsiteIntent"}, "context": null}
2016-12-24 11:38:33,980 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "call", "end": "DialCallKeyword"}, "context": null}
2016-12-24 11:38:33,983 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "phone", "end": "DialCallKeyword"}, "context": null}
2016-12-24 11:38:33,987 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(call|phone) (?P<Contact>.*)"}, "context": null}
2016-12-24 11:38:33,994 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["DialCallKeyword", "DialCallKeyword"], ["Contact", "Contact"]], "optional": [], "name": "DialCallIntent"}, "context": null}

2016-12-24 11:38:34,000 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hello world", "end": "HelloWorldKeyword"}, "context": null}
2016-12-24 11:38:34,003 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "greetings", "end": "HelloWorldKeyword"}, "context": null}
2016-12-24 11:38:34,006 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how are you", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,009 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how have you been", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,011 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how has your day been", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,018 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thank you", "end": "ThankYouKeyword"}, "context": null}
2016-12-24 11:38:34,023 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thanks", "end": "ThankYouKeyword"}, "context": null}
2016-12-24 11:38:34,027 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ThankYouKeyword", "ThankYouKeyword"]], "optional": [], "name": "ThankYouIntent"}, "context": null}
2016-12-24 11:38:34,036 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["HowAreYouKeyword", "HowAreYouKeyword"]], "optional": [], "name": "HowAreYouIntent"}, "context": null}
2016-12-24 11:38:34,041 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["HelloWorldKeyword", "HelloWorldKeyword"]], "optional": [], "name": "HelloWorldIntent"}, "context": null}
2016-12-24 11:38:34,046 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "IP address", "end": "IPCommand"}, "context": null}
2016-12-24 11:38:34,050 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "IP", "end": "IPCommand"}, "context": null}
2016-12-24 11:38:34,053 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network address", "end": "IPCommand"}, "context": null}
2016-12-24 11:38:34,061 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["IPCommand", "IPCommand"]], "optional": [], "name": "IPIntent"}, "context": null}
2016-12-24 11:38:34,066 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "joke", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:38:34,071 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "make me laugh", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:38:34,075 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "brighten my day", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:38:34,083 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me a joke", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:38:34,088 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["JokingKeyword", "JokingKeyword"]], "optional": [], "name": "JokingIntent"}, "context": null}
2016-12-24 11:38:34,090 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "go to sleep", "end": "SleepCommand"}, "context": null}
2016-12-24 11:38:34,096 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nap time", "end": "SleepCommand"}, "context": null}
2016-12-24 11:38:34,101 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleepy time tea", "end": "SleepCommand"}, "context": null}
2016-12-24 11:38:34,113 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleepytime tea", "end": "SleepCommand"}, "context": null}
2016-12-24 11:38:34,120 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SleepCommand", "SleepCommand"]], "optional": [], "name": "NapTimeIntent"}, "context": null}
2016-12-24 11:38:34,123 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "news", "end": "NPRNewsKeyword"}, "context": null}
2016-12-24 11:38:34,134 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me the news", "end": "NPRNewsKeyword"}, "context": null}
2016-12-24 11:38:34,140 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["NPRNewsKeyword", "NPRNewsKeyword"]], "optional": [], "name": "NPRNewsIntent"}, "context": null}
2016-12-24 11:38:34,143 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,150 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "temperature", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,153 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "forecast", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,157 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rain", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,174 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raining", "end": "WeatherType", "alias_of": "rain"}, "context": null}
2016-12-24 11:38:34,183 - Skills - DEBUG - {"type": "enclosure.system.reset", "data": {}, "context": null}
2016-12-24 11:38:34,185 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snow", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,201 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snowing", "end": "WeatherType", "alias_of": "snow"}, "context": null}
2016-12-24 11:38:34,207 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleet", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,216 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hail", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,220 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hailing", "end": "WeatherType", "alias_of": "hail"}, "context": null}
2016-12-24 11:38:34,233 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sunny", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,237 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sun", "end": "WeatherType", "alias_of": "sunny"}, "context": null}
2016-12-24 11:38:34,239 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hot", "end": "WeatherType"}, "context": null}

2016-12-24 11:38:34,261 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "warm", "end": "WeatherType", "alias_of": "hot"}, "context": null}
2016-12-24 11:38:34,267 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cold", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,285 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cool", "end": "WeatherType", "alias_of": "cold"}, "context": null}
2016-12-24 11:38:34,293 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,301 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tomorrow", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,314 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,317 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in 1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,319 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,329 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in one day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,331 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,337 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,340 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "following day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,347 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:34,362 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Seattle", "end": "Location"}, "context": null}
2016-12-24 11:38:34,367 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Los Angeles california", "end": "Location"}, "context": null}
2016-12-24 11:38:34,370 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "los angeles", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:38:34,377 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "la", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:38:34,381 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Lawrence kansas", "end": "Location"}, "context": null}
2016-12-24 11:38:34,394 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland oregon", "end": "Location"}, "context": null}
2016-12-24 11:38:34,399 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland", "end": "Location", "alias_of": "portland oregon"}, "context": null}
2016-12-24 11:38:34,408 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spokane", "end": "Location"}, "context": null}
2016-12-24 11:38:34,414 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "new york", "end": "Location"}, "context": null}
2016-12-24 11:38:34,416 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chicago", "end": "Location"}, "context": null}
2016-12-24 11:38:34,421 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "houston", "end": "Location"}, "context": null}
2016-12-24 11:38:34,423 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "austin", "end": "Location"}, "context": null}
2016-12-24 11:38:34,430 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san diego", "end": "Location"}, "context": null}
2016-12-24 11:38:34,435 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san francisco", "end": "Location"}, "context": null}
2016-12-24 11:38:34,440 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san jose", "end": "Location"}, "context": null}
2016-12-24 11:38:34,445 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boston", "end": "Location"}, "context": null}
2016-12-24 11:38:34,450 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "washington d c", "end": "Location"}, "context": null}
2016-12-24 11:38:34,453 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atlanta", "end": "Location"}, "context": null}
2016-12-24 11:38:34,465 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,471 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,482 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,493 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,500 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,502 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,504 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,509 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,512 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,514 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,521 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,527 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next couple of hours", "end": "NextHours"}, "context": null}

2016-12-24 11:38:34,531 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,538 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:34,543 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:38:34,550 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pair", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:38:34,554 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pairing", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:38:34,556 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "register", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:38:34,563 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "unit", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:38:34,566 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "device", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:38:34,569 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mycroft", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:38:34,580 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["PairingKeyword", "PairingKeyword"], ["DeviceKeyword", "DeviceKeyword"]], "optional": [], "name": "PairingIntent"}, "context": null}
2016-12-24 11:38:34,583 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where were you born", "end": "WhereWereYouBornKeyword"}, "context": null}
2016-12-24 11:38:34,587 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where were you created", "end": "WhereWereYouBornKeyword"}, "context": null}
2016-12-24 11:38:34,589 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "when were you born", "end": "WhenWereYouBornKeyword"}, "context": null}
2016-12-24 11:38:34,598 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "when were you created", "end": "WhenWereYouBornKeyword"}, "context": null}
2016-12-24 11:38:34,603 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who made you", "end": "WhoMadeYouKeyword"}, "context": null}
2016-12-24 11:38:34,607 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who were you made by", "end": "WhoMadeYouKeyword"}, "context": null}
2016-12-24 11:38:34,615 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what are you", "end": "WhatAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,622 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who are you", "end": "WhoAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,625 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who're you", "end": "WhoAreYouKeyword"}, "context": null}
2016-12-24 11:38:34,632 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhenWereYouBornKeyword", "WhenWereYouBornKeyword"]], "optional": [], "name": "WhenWereYouBornIntent"}, "context": null}
2016-12-24 11:38:34,635 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhereWereYouBornKeyword", "WhereWereYouBornKeyword"]], "optional": [], "name": "WhereWereYouBornIntent"}, "context": null}
2016-12-24 11:38:34,642 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhoMadeYouKeyWord", "WhoMadeYouKeyWord"]], "optional": [], "name": "WhoMadeYouIntent"}, "context": null}
2016-12-24 11:38:34,657 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhoAreYouKeyword", "WhoAreYouKeyword"]], "optional": [], "name": "WhoAreYouIntent"}, "context": null}
2016-12-24 11:38:34,665 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhatAreYouKeyword", "WhatAreYouKeyword"]], "optional": [], "name": "WhatAreYouIntent"}, "context": null}
2016-12-24 11:38:34,668 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:38:34,672 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all my", "end": "ReminderSkillAmount", "alias_of": "all"}, "context": null}
2016-12-24 11:38:34,674 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:38:34,678 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "ReminderSkillAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:38:34,681 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:38:34,687 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "ReminderSkillAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:38:34,690 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:38:34,700 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:38:34,702 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "show", "end": "ReminderSkillListVerb"}, "context": null}
2016-12-24 11:38:34,704 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "list", "end": "ReminderSkillListVerb"}, "context": null}
2016-12-24 11:38:34,707 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "erase", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:34,711 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:34,714 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "delete", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:34,718 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remove", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:38:34,720 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminder", "end": "ReminderSkillKeyword"}, "context": null}

2016-12-24 11:38:34,722 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminders", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:38:34,724 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notification", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:38:34,726 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notifications", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:38:34,728 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "off", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:38:34,730 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:38:34,737 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:38:34,745 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remind", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,747 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notify", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,749 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notify me", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,753 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remind me", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,757 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminder", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,759 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "set a reminder", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:38:34,763 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<ReminderSkillAmount>\\d+)"}, "context": null}
2016-12-24 11:38:34,767 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillCreateVerb", "ReminderSkillCreateVerb"]], "optional": [], "name": "ReminderSkillCreateIntent"}, "context": null}
2016-12-24 11:38:34,771 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillListVerb", "ReminderSkillListVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [["ReminderSkillAmount", "ReminderSkillAmount"]], "name": "ReminderSkillListIntent"}, "context": null}
2016-12-24 11:38:34,775 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillDeleteVerb", "ReminderSkillDeleteVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [["ReminderSkillAmount", "ReminderSkillAmount"]], "name": "ReminderSkillDeleteIntent"}, "context": null}
2016-12-24 11:38:34,778 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillStopVerb", "ReminderSkillStopVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [], "name": "ReminderSkillStopIntent"}, "context": null}
2016-12-24 11:38:34,780 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "speak", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:38:34,784 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "say", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:38:34,788 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "repeat", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:38:34,791 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "speak (?P<Words>.*)"}, "context": null}
2016-12-24 11:38:34,795 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "say (?P<Words>.*)"}, "context": null}
2016-12-24 11:38:34,798 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "repeat (?P<Words>.*)"}, "context": null}
2016-12-24 11:38:34,807 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SpeakKeyword", "SpeakKeyword"], ["Words", "Words"]], "optional": [], "name": "SpeakIntent"}, "context": null}
2016-12-24 11:38:34,811 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spell", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:38:34,814 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spell the word", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:38:34,818 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spelling of", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:38:34,822 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spelling of the word", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:38:34,831 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(spell the word|spelling of the word|spelling of|spell) (?P<Word>\\w+)"}, "context": null}
2016-12-24 11:38:34,834 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SpellingKeyword", "SpellingKeyword"], ["Word", "Word"]], "optional": [], "name": "SpellingIntent"}, "context": null}
2016-12-24 11:38:34,836 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stock price", "end": "StockPriceKeyword"}, "context": null}
2016-12-24 11:38:34,843 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "trading at", "end": "StockPriceKeyword"}, "context": null}
2016-12-24 11:38:34,846 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "price of (?P<Company>.*)"}, "context": null}
2016-12-24 11:38:34,849 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "is (?P<Company>.*) trading at"}, "context": null}
2016-12-24 11:38:34,853 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["StockPriceKeyword", "StockPriceKeyword"], ["Company", "Company"]], "optional": [], "name": "StockPriceIntent"}, "context": null}
2016-12-24 11:38:34,855 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "StopKeyword"}, "context": null}
2016-12-24 11:38:34,866 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "silence", "end": "StopKeyword"}, "context": null}

2016-12-24 11:38:34,869 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "shut up", "end": "StopKeyword"}, "context": null}
2016-12-24 11:38:34,872 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "be quiet", "end": "StopKeyword"}, "context": null}
2016-12-24 11:38:34,878 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["StopKeyword", "StopKeyword"]], "optional": [], "name": "StopIntent"}, "context": null}
2016-12-24 11:38:34,890 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reset volume", "end": "ResetVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,893 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "restart volume", "end": "ResetVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,895 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lower volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,897 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reduce volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,902 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "decrease volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,914 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "0", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,919 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "zero", "end": "VolumeAmount", "alias_of": "0"}, "context": null}
2016-12-24 11:38:34,922 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,925 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "VolumeAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:38:34,928 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,931 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "VolumeAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:38:34,933 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "quiet", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,935 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "normal", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,939 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "loud", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:38:34,941 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mute volume", "end": "MuteVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,945 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "volume", "end": "VolumeKeyword"}, "context": null}
2016-12-24 11:38:34,948 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rise volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,952 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raise volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,955 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boost volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,958 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "increase volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:38:34,961 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<VolumeAmount>\\d+)"}, "context": null}
2016-12-24 11:38:34,964 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["VolumeKeyword", "VolumeKeyword"], ["VolumeAmount", "VolumeAmount"]], "optional": [], "name": "SetVolumeIntent"}, "context": null}
2016-12-24 11:38:34,967 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["IncreaseVolumeKeyword", "IncreaseVolumeKeyword"]], "optional": [], "name": "IncreaseVolumeIntent"}, "context": null}
2016-12-24 11:38:34,971 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["DecreaseVolumeKeyword", "DecreaseVolumeKeyword"]], "optional": [], "name": "DecreaseVolumeIntent"}, "context": null}
2016-12-24 11:38:34,974 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["MuteVolumeKeyword", "MuteVolumeKeyword"]], "optional": [], "name": "MuteVolumeIntent"}, "context": null}
2016-12-24 11:38:34,976 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ResetVolumeKeyword", "ResetVolumeKeyword"]], "optional": [], "name": "ResetVolumeIntent"}, "context": null}
2016-12-24 11:38:34,979 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,981 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "temperature", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,983 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "forecast", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:38:34,997 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rain", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:34,999 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raining", "end": "WeatherType", "alias_of": "rain"}, "context": null}
2016-12-24 11:38:35,001 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snow", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,003 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snowing", "end": "WeatherType", "alias_of": "snow"}, "context": null}
2016-12-24 11:38:35,007 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleet", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,010 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hail", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,014 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hailing", "end": "WeatherType", "alias_of": "hail"}, "context": null}

2016-12-24 11:38:35,016 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sunny", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,019 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sun", "end": "WeatherType", "alias_of": "sunny"}, "context": null}
2016-12-24 11:38:35,021 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hot", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,024 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "warm", "end": "WeatherType", "alias_of": "hot"}, "context": null}
2016-12-24 11:38:35,027 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cold", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,029 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cool", "end": "WeatherType", "alias_of": "cold"}, "context": null}
2016-12-24 11:38:35,032 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "", "end": "WeatherType"}, "context": null}
2016-12-24 11:38:35,035 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tomorrow", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,037 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,042 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in 1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,044 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,048 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in one day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,052 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,056 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,059 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "following day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,062 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following day", "end": "NextDay"}, "context": null}
2016-12-24 11:38:35,066 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Seattle", "end": "Location"}, "context": null}
2016-12-24 11:38:35,069 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Los Angeles california", "end": "Location"}, "context": null}
2016-12-24 11:38:35,073 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "los angeles", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:38:35,077 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "la", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:38:35,080 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Lawrence kansas", "end": "Location"}, "context": null}
2016-12-24 11:38:35,082 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland oregon", "end": "Location"}, "context": null}
2016-12-24 11:38:35,084 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland", "end": "Location", "alias_of": "portland oregon"}, "context": null}
2016-12-24 11:38:35,085 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spokane", "end": "Location"}, "context": null}
2016-12-24 11:38:35,090 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "new york", "end": "Location"}, "context": null}
2016-12-24 11:38:35,092 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chicago", "end": "Location"}, "context": null}
2016-12-24 11:38:35,096 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "houston", "end": "Location"}, "context": null}
2016-12-24 11:38:35,099 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "austin", "end": "Location"}, "context": null}
2016-12-24 11:38:35,102 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san diego", "end": "Location"}, "context": null}
2016-12-24 11:38:35,105 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san francisco", "end": "Location"}, "context": null}
2016-12-24 11:38:35,108 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san jose", "end": "Location"}, "context": null}
2016-12-24 11:38:35,110 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boston", "end": "Location"}, "context": null}
2016-12-24 11:38:35,113 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "washington d c", "end": "Location"}, "context": null}
2016-12-24 11:38:35,115 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atlanta", "end": "Location"}, "context": null}
2016-12-24 11:38:35,118 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,120 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,123 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,127 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,129 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,131 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,135 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,138 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,141 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next few hours", "end": "NextHours"}, "context": null}

2016-12-24 11:38:35,144 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,146 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,149 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,151 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,155 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:38:35,157 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:38:35,160 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"]], "optional": [["Location", "Location"]], "name": "CurrentWeatherIntent"}, "context": null}
2016-12-24 11:38:35,164 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"], ["NextHours", "NextHours"]], "optional": [["Location", "Location"]], "name": "NextHoursWeatherIntent"}, "context": null}
2016-12-24 11:38:35,167 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"], ["NextDay", "NextDay"]], "optional": [["Location", "Location"]], "name": "NextDayWeatherIntent"}, "context": null}
2016-12-24 11:38:35,169 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wiki", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:38:35,172 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wikipedia", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:38:35,175 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:38:35,178 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell us about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:38:35,180 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what does wikipedia say about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:38:35,182 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(tell us about|tell me about|what does wikipedia say about|wiki|wikipedia) (?P<ArticleTitle>.*)"}, "context": null}
2016-12-24 11:38:35,184 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WikipediaKeyword", "WikipediaKeyword"], ["ArticleTitle", "ArticleTitle"]], "optional": [], "name": "WikipediaIntent"}, "context": null}
2016-12-24 11:38:36,468 - Skills - DEBUG - {"type": "recognizer_loop:utterance", "data": {"utterances": ["what tim"]}, "context": null}
2016-12-24 11:38:36,490 - intent__init__ - ERROR - Traceback (most recent call last):
  File "/home/paul/Irene/mycroft/skills/intent/__init__.py", line 48, in handle_utterance
    utterance, 100))
StopIteration
2016-12-24 11:38:36,520 - Skills - DEBUG - {"type": "intent_failure", "data": {"utterance": "what tim"}, "context": null}
2016-12-24 11:38:36,524 - wolfram_alpha__init__ - DEBUG - Could not determine intent, falling back to WolframAlpha Skill!
2016-12-24 11:38:36,536 - Skills - DEBUG - {"type": "enclosure.mouth.think", "data": {}, "context": null}
2016-12-24 11:38:36,539 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:38:42,239 - requests.packages.urllib3.connectionpool - DEBUG - "GET /v1/wa?input=what+tim HTTP/1.1" 200 5531
2016-12-24 11:38:42,299 - Skills - DEBUG - {"type": "speak", "data": {"utterance": "Tim (movie) : title, Tim, director, Michael Pate, release date, September 17, 1981 (35 years 3 months ago), runtime, 109 minutes (1 hour 49 minutes), writers, Colleen McCullough, Michael Pate, genres, drama, romance"}, "context": null}
2016-12-24 11:38:42,336 - Skills - DEBUG - {"type": "recognizer_loop:audio_output_start", "data": {}, "context": null}
2016-12-24 11:38:44,954 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:45,178 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:45,277 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:45,325 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:45,377 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:45,460 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:38:45,523 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "5"}, "context": null}
2016-12-24 11:38:45,600 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:45,792 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:45,993 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:46,091 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:46,238 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:46,248 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:46,363 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}

2016-12-24 11:38:46,442 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:46,558 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:46,645 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:46,815 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:46,889 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:46,996 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:47,023 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:47,051 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:38:47,158 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:47,254 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:47,344 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:38:47,433 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:38:47,708 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:47,909 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:38:47,953 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:38:48,045 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
/home/paul/Irene/init/start.sh: line 26: 28980 Terminated PYTHONPATH=${TOP} python ${SCRIPT} $@
Starting skills
2016-12-24 11:40:15,944 - mycroft.configuration - DEBUG - Configuration '/home/paul/Irene/mycroft/configuration/mycroft.conf' loaded
2016-12-24 11:40:15,945 - mycroft.configuration - DEBUG - Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:40:15,945 - mycroft.configuration - DEBUG - Configuration '/home/paul/.mycroft/mycroft.conf' not found
2016-12-24 11:40:16,256 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:40:17,598 - requests.packages.urllib3.connectionpool - DEBUG - "GET /v1/device/c14b1936-4458-4859-9cab-ebb56da21994/setting HTTP/1.1" 200 2887
2016-12-24 11:40:17,755 - mycroft.messagebus.client.ws - INFO - Connected

** (process:29476): [1;33mWARNING[0m **: Trying to register gtype 'GMountMountFlags' as enum when in fact it is of type 'GFlags'

** (process:29476): [1;33mWARNING[0m **: Trying to register gtype 'GDriveStartFlags' as enum when in fact it is of type 'GFlags'

** (process:29476): [1;33mWARNING[0m **: Trying to register gtype 'GSocketMsgFlags' as enum when in fact it is of type 'GFlags'
2016-12-24 11:40:21,086 - Skills - DEBUG - {"type": "connected", "data": {}, "context": null}
2016-12-24 11:40:21,095 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timer", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:40:21,097 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timers", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:40:21,104 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarm", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:40:21,106 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarms", "end": "AlarmSkillCreateVerb"}, "context": null}
2016-12-24 11:40:21,110 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:40:21,112 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all my", "end": "AlarmSkillAmount", "alias_of": "all"}, "context": null}
2016-12-24 11:40:21,114 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:40:21,120 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "AlarmSkillAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:40:21,123 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:40:21,125 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "AlarmSkillAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:40:21,127 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:40:21,131 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following", "end": "AlarmSkillAmount"}, "context": null}
2016-12-24 11:40:21,134 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "off", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,136 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,138 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "AlarmSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,148 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "erase", "end": "AlarmSkillDeleteVerb"}, "context": null}

2016-12-24 11:40:21,152 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:21,181 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "delete", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:21,183 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remove", "end": "AlarmSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:21,190 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timer", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:40:21,193 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "timers", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:40:21,196 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarm", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:40:21,199 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "alarms", "end": "AlarmSkillKeyword"}, "context": null}
2016-12-24 11:40:21,204 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "show", "end": "AlarmSkillListVerb"}, "context": null}
2016-12-24 11:40:21,206 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "list", "end": "AlarmSkillListVerb"}, "context": null}
2016-12-24 11:40:21,215 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<AlarmSkillAmount>\\d+)"}, "context": null}
2016-12-24 11:40:21,268 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillCreateVerb", "AlarmSkillCreateVerb"]], "optional": [], "name": "AlarmSkillCreateIntent"}, "context": null}
2016-12-24 11:40:21,279 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillListVerb", "AlarmSkillListVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [["AlarmSkillAmount", "AlarmSkillAmount"]], "name": "AlarmSkillListIntent"}, "context": null}
2016-12-24 11:40:21,289 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillDeleteVerb", "AlarmSkillDeleteVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [["AlarmSkillAmount", "AlarmSkillAmount"]], "name": "AlarmSkillDeleteIntent"}, "context": null}
2016-12-24 11:40:21,293 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AlarmSkillStopVerb", "AlarmSkillStopVerb"], ["AlarmSkillKeyword", "AlarmSkillKeyword"]], "optional": [], "name": "AlarmSkillStopIntent"}, "context": null}
2016-12-24 11:40:21,301 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "record", "end": "AudioRecordSkillKeyword"}, "context": null}
2016-12-24 11:40:21,312 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "recording", "end": "AudioRecordSkillKeyword"}, "context": null}
2016-12-24 11:40:21,324 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,341 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,354 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "AudioRecordSkillStopVerb"}, "context": null}
2016-12-24 11:40:21,363 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,373 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "play", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,383 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "replay", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,394 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "playback", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,402 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reproduce", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,406 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "running", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,417 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "playing", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,423 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "replaying", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,428 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reproducing", "end": "AudioRecordSkillPlayVerb"}, "context": null}
2016-12-24 11:40:21,437 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillIntent"}, "context": null}
2016-12-24 11:40:21,447 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillStopVerb", "AudioRecordSkillStopVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillStopIntent"}, "context": null}
2016-12-24 11:40:21,454 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillPlayVerb", "AudioRecordSkillPlayVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillPlayIntent"}, "context": null}
2016-12-24 11:40:21,469 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["AudioRecordSkillStopVerb", "AudioRecordSkillStopVerb"], ["AudioRecordSkillPlayVerb", "AudioRecordSkillPlayVerb"], ["AudioRecordSkillKeyword", "AudioRecordSkillKeyword"]], "optional": [], "name": "AudioRecordSkillStopPlayIntent"}, "context": null}
2016-12-24 11:40:21,473 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "config", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:40:21,480 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "configuration", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:40:21,485 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "setting", "end": "ConfigurationSkillKeyword"}, "context": null}
2016-12-24 11:40:21,496 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "update", "end": "ConfigurationSkillUpdateVerb"}, "context": null}

2016-12-24 11:40:21,503 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reload", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:40:21,510 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sync", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:40:21,517 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "synchronize", "end": "ConfigurationSkillUpdateVerb"}, "context": null}
2016-12-24 11:40:21,531 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ConfigurationSkillKeyword", "ConfigurationSkillKeyword"], ["ConfigurationSkillUpdateVerb", "ConfigurationSkillUpdateVerb"]], "optional": [], "name": "UpdateConfigurationIntent"}, "context": null}
2016-12-24 11:40:21,540 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what time is it", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,552 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what is the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,561 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what's the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,567 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "whats the time", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,581 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what time is", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,589 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "time is it", "end": "TimeKeyword"}, "context": null}
2016-12-24 11:40:21,593 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what's the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:40:21,604 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "whats the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:40:21,607 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what day is it", "end": "DateKeyword"}, "context": null}
2016-12-24 11:40:21,613 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what is the date", "end": "DateKeyword"}, "context": null}
2016-12-24 11:40:21,621 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:40:21,628 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["TimeKeyword", "TimeKeyword"]], "optional": [["Location", "Location"]], "name": "TimeIntent"}, "context": null}
2016-12-24 11:40:21,638 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "SearchKeyword"}, "context": null}
2016-12-24 11:40:21,640 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "find", "end": "SearchKeyword"}, "context": null}
2016-12-24 11:40:21,646 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "google", "end": "Website"}, "context": null}
2016-12-24 11:40:21,650 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "facebook", "end": "Website"}, "context": null}
2016-12-24 11:40:21,657 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "amazon", "end": "Website"}, "context": null}
2016-12-24 11:40:21,664 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "youtube", "end": "Website"}, "context": null}
2016-12-24 11:40:21,668 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "yahoo", "end": "Website"}, "context": null}
2016-12-24 11:40:21,672 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wikipedia", "end": "Website"}, "context": null}
2016-12-24 11:40:21,683 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "ebay", "end": "Website"}, "context": null}
2016-12-24 11:40:21,688 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "twitter", "end": "Website"}, "context": null}
2016-12-24 11:40:21,691 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "go", "end": "Website"}, "context": null}
2016-12-24 11:40:21,696 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "craigslist", "end": "Website"}, "context": null}
2016-12-24 11:40:21,701 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reddit", "end": "Website"}, "context": null}
2016-12-24 11:40:21,704 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "linkedin", "end": "Website"}, "context": null}
2016-12-24 11:40:21,708 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "netflix", "end": "Website"}, "context": null}
2016-12-24 11:40:21,711 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "live", "end": "Website"}, "context": null}
2016-12-24 11:40:21,715 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bing", "end": "Website"}, "context": null}
2016-12-24 11:40:21,720 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pinterest", "end": "Website"}, "context": null}
2016-12-24 11:40:21,724 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "espn", "end": "Website"}, "context": null}
2016-12-24 11:40:21,728 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "imgur", "end": "Website"}, "context": null}
2016-12-24 11:40:21,737 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tumblr", "end": "Website"}, "context": null}
2016-12-24 11:40:21,748 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chase", "end": "Website"}, "context": null}
2016-12-24 11:40:21,752 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cnn", "end": "Website"}, "context": null}
2016-12-24 11:40:21,756 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "paypal", "end": "Website"}, "context": null}
2016-12-24 11:40:21,759 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "instagram", "end": "Website"}, "context": null}
2016-12-24 11:40:21,763 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "blogspot", "end": "Website"}, "context": null}

2016-12-24 11:40:21,766 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "apple", "end": "Website"}, "context": null}
2016-12-24 11:40:21,771 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "launch", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:40:21,776 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "open", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:40:21,780 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "LaunchKeyword"}, "context": null}
2016-12-24 11:40:21,795 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "for (?P<SearchTerms>.*)"}, "context": null}
2016-12-24 11:40:21,804 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "for (?P<SearchTerms>.*) on"}, "context": null}
2016-12-24 11:40:21,808 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<SearchTerms>.*) on"}, "context": null}
2016-12-24 11:40:21,813 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cd/dvd creator", "end": "Application"}, "context": null}
2016-12-24 11:40:21,818 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cd", "end": "Application"}, "context": null}
2016-12-24 11:40:21,822 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse & touch\u00adpad", "end": "Application"}, "context": null}
2016-12-24 11:40:21,827 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse", "end": "Application"}, "context": null}
2016-12-24 11:40:21,834 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bleachbit", "end": "Application"}, "context": null}
2016-12-24 11:40:21,840 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "assistive technologies", "end": "Application"}, "context": null}
2016-12-24 11:40:21,844 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "assistive", "end": "Application"}, "context": null}
2016-12-24 11:40:21,850 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "glade", "end": "Application"}, "context": null}
2016-12-24 11:40:21,863 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome system monitor", "end": "Application"}, "context": null}
2016-12-24 11:40:21,869 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:40:21,872 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "help", "end": "Application"}, "context": null}
2016-12-24 11:40:21,876 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop search", "end": "Application"}, "context": null}
2016-12-24 11:40:21,882 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:40:21,885 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cheese", "end": "Application"}, "context": null}
2016-12-24 11:40:21,889 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "customize look and feel", "end": "Application"}, "context": null}
2016-12-24 11:40:21,895 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "customize", "end": "Application"}, "context": null}
2016-12-24 11:40:21,900 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "feh", "end": "Application"}, "context": null}
2016-12-24 11:40:21,912 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "robots", "end": "Application"}, "context": null}
2016-12-24 11:40:21,920 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "home folder", "end": "Application"}, "context": null}
2016-12-24 11:40:21,945 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "home", "end": "Application"}, "context": null}
2016-12-24 11:40:21,954 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "brasero", "end": "Application"}, "context": null}
2016-12-24 11:40:21,970 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gparted", "end": "Application"}, "context": null}
2016-12-24 11:40:21,994 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:21,997 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font", "end": "Application"}, "context": null}
2016-12-24 11:40:22,003 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pri\u00adva\u00adcy", "end": "Application"}, "context": null}
2016-12-24 11:40:22,006 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mines", "end": "Application"}, "context": null}
2016-12-24 11:40:22,011 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome shell extension preferences", "end": "Application"}, "context": null}
2016-12-24 11:40:22,016 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:40:22,020 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "image viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,025 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "image", "end": "Application"}, "context": null}
2016-12-24 11:40:22,028 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mutter", "end": "Application"}, "context": null}
2016-12-24 11:40:22,033 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:40:22,037 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thunderbird", "end": "Application"}, "context": null}
2016-12-24 11:40:22,042 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,048 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about", "end": "Application"}, "context": null}
2016-12-24 11:40:22,062 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firefox", "end": "Application"}, "context": null}
2016-12-24 11:40:22,066 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}

2016-12-24 11:40:22,075 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice writer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,078 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,081 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "prin\u00adters", "end": "Application"}, "context": null}
2016-12-24 11:40:22,090 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth manager", "end": "Application"}, "context": null}
2016-12-24 11:40:22,097 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:40:22,102 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "co\u00adlor", "end": "Application"}, "context": null}
2016-12-24 11:40:22,106 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:40:22,109 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "no\u00adti\u00adfi\u00adca\u00adtions", "end": "Application"}, "context": null}
2016-12-24 11:40:22,114 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mplayer media player", "end": "Application"}, "context": null}
2016-12-24 11:40:22,125 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mplayer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,131 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tweak tool", "end": "Application"}, "context": null}
2016-12-24 11:40:22,134 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tweak", "end": "Application"}, "context": null}
2016-12-24 11:40:22,144 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "users", "end": "Application"}, "context": null}
2016-12-24 11:40:22,148 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate dictionary", "end": "Application"}, "context": null}
2016-12-24 11:40:22,153 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,156 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "take screenshot", "end": "Application"}, "context": null}
2016-12-24 11:40:22,158 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "take", "end": "Application"}, "context": null}
2016-12-24 11:40:22,162 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake preferences", "end": "Application"}, "context": null}
2016-12-24 11:40:22,167 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake", "end": "Application"}, "context": null}
2016-12-24 11:40:22,172 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "file management", "end": "Application"}, "context": null}
2016-12-24 11:40:22,174 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "file", "end": "Application"}, "context": null}
2016-12-24 11:40:22,176 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hdspconf", "end": "Application"}, "context": null}
2016-12-24 11:40:22,178 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "polari", "end": "Application"}, "context": null}
2016-12-24 11:40:22,183 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice math", "end": "Application"}, "context": null}
2016-12-24 11:40:22,187 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,194 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice draw", "end": "Application"}, "context": null}
2016-12-24 11:40:22,201 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,204 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "characters", "end": "Application"}, "context": null}
2016-12-24 11:40:22,208 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "windows", "end": "Application"}, "context": null}
2016-12-24 11:40:22,212 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:40:22,215 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chess", "end": "Application"}, "context": null}
2016-12-24 11:40:22,220 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "web", "end": "Application"}, "context": null}
2016-12-24 11:40:22,225 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color profile viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,228 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:40:22,234 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about me", "end": "Application"}, "context": null}
2016-12-24 11:40:22,236 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "about", "end": "Application"}, "context": null}
2016-12-24 11:40:22,240 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice xslt based filters", "end": "Application"}, "context": null}
2016-12-24 11:40:22,243 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,248 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "autorun prompt", "end": "Application"}, "context": null}
2016-12-24 11:40:22,251 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "autorun", "end": "Application"}, "context": null}
2016-12-24 11:40:22,257 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "passwords and keys", "end": "Application"}, "context": null}
2016-12-24 11:40:22,260 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "passwords", "end": "Application"}, "context": null}
2016-12-24 11:40:22,263 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnu image manipulation program", "end": "Application"}, "context": null}

2016-12-24 11:40:22,265 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnu", "end": "Application"}, "context": null}
2016-12-24 11:40:22,268 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "klotski", "end": "Application"}, "context": null}
2016-12-24 11:40:22,274 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pictures folder", "end": "Application"}, "context": null}
2016-12-24 11:40:22,278 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pictures", "end": "Application"}, "context": null}
2016-12-24 11:40:22,287 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dictionary", "end": "Application"}, "context": null}
2016-12-24 11:40:22,291 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mahjongg", "end": "Application"}, "context": null}
2016-12-24 11:40:22,294 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "log file viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,298 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "log", "end": "Application"}, "context": null}
2016-12-24 11:40:22,301 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution calendar", "end": "Application"}, "context": null}
2016-12-24 11:40:22,304 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution", "end": "Application"}, "context": null}
2016-12-24 11:40:22,307 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sysprof", "end": "Application"}, "context": null}
2016-12-24 11:40:22,310 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "theme installer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,315 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "theme", "end": "Application"}, "context": null}
2016-12-24 11:40:22,319 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate color selection", "end": "Application"}, "context": null}
2016-12-24 11:40:22,322 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,324 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cmake", "end": "Application"}, "context": null}
2016-12-24 11:40:22,327 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dis\u00adplays", "end": "Application"}, "context": null}
2016-12-24 11:40:22,332 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard shortcuts", "end": "Application"}, "context": null}
2016-12-24 11:40:22,335 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:40:22,337 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "net\u00adwork", "end": "Application"}, "context": null}
2016-12-24 11:40:22,342 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cosmos", "end": "Application"}, "context": null}
2016-12-24 11:40:22,344 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "quadrapassel", "end": "Application"}, "context": null}
2016-12-24 11:40:22,348 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mouse", "end": "Application"}, "context": null}
2016-12-24 11:40:22,352 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "control center", "end": "Application"}, "context": null}
2016-12-24 11:40:22,355 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "control", "end": "Application"}, "context": null}
2016-12-24 11:40:22,358 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sha\u00adring", "end": "Application"}, "context": null}
2016-12-24 11:40:22,362 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "computer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,371 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "view file", "end": "Application"}, "context": null}
2016-12-24 11:40:22,374 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "view", "end": "Application"}, "context": null}
2016-12-24 11:40:22,377 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating gnome", "end": "Application"}, "context": null}
2016-12-24 11:40:22,380 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating", "end": "Application"}, "context": null}
2016-12-24 11:40:22,383 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network tools", "end": "Application"}, "context": null}
2016-12-24 11:40:22,385 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:40:22,387 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "eye of mate image viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,388 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "eye", "end": "Application"}, "context": null}
2016-12-24 11:40:22,390 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote desktop viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,396 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote", "end": "Application"}, "context": null}
2016-12-24 11:40:22,398 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "Application"}, "context": null}
2016-12-24 11:40:22,402 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "accerciser", "end": "Application"}, "context": null}
2016-12-24 11:40:22,406 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rhythmbox", "end": "Application"}, "context": null}
2016-12-24 11:40:22,409 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qt v4l2 test utility", "end": "Application"}, "context": null}
2016-12-24 11:40:22,412 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qt", "end": "Application"}, "context": null}
2016-12-24 11:40:22,415 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "maps", "end": "Application"}, "context": null}
2016-12-24 11:40:22,419 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power statistics", "end": "Application"}, "context": null}

2016-12-24 11:40:22,423 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power", "end": "Application"}, "context": null}
2016-12-24 11:40:22,426 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2048", "end": "Application"}, "context": null}
2016-12-24 11:40:22,428 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nitrogen", "end": "Application"}, "context": null}
2016-12-24 11:40:22,431 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "panel", "end": "Application"}, "context": null}
2016-12-24 11:40:22,434 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "iagno", "end": "Application"}, "context": null}
2016-12-24 11:40:22,437 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "messenger for desktop", "end": "Application"}, "context": null}
2016-12-24 11:40:22,440 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "messenger", "end": "Application"}, "context": null}
2016-12-24 11:40:22,442 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "archive manager", "end": "Application"}, "context": null}
2016-12-24 11:40:22,446 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "archive", "end": "Application"}, "context": null}
2016-12-24 11:40:22,449 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "root terminal", "end": "Application"}, "context": null}
2016-12-24 11:40:22,452 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "root", "end": "Application"}, "context": null}
2016-12-24 11:40:22,455 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome mplayer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,457 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:40:22,460 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color picker", "end": "Application"}, "context": null}
2016-12-24 11:40:22,462 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:40:22,465 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pulseaudio volume control", "end": "Application"}, "context": null}
2016-12-24 11:40:22,468 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pulseaudio", "end": "Application"}, "context": null}
2016-12-24 11:40:22,470 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi vnc server browser", "end": "Application"}, "context": null}
2016-12-24 11:40:22,475 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:40:22,478 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firewall configuration", "end": "Application"}, "context": null}
2016-12-24 11:40:22,483 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "firewall", "end": "Application"}, "context": null}
2016-12-24 11:40:22,486 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pop art squares", "end": "Application"}, "context": null}
2016-12-24 11:40:22,489 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pop", "end": "Application"}, "context": null}
2016-12-24 11:40:22,492 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hdspmixer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,502 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "screensaver", "end": "Application"}, "context": null}
2016-12-24 11:40:22,505 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "settings", "end": "Application"}, "context": null}
2016-12-24 11:40:22,508 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "electron", "end": "Application"}, "context": null}
2016-12-24 11:40:22,511 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm community edition", "end": "Application"}, "context": null}
2016-12-24 11:40:22,515 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm", "end": "Application"}, "context": null}
2016-12-24 11:40:22,519 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hitori", "end": "Application"}, "context": null}
2016-12-24 11:40:22,521 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thunderbird", "end": "Application"}, "context": null}
2016-12-24 11:40:22,524 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,527 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "floating", "end": "Application"}, "context": null}
2016-12-24 11:40:22,530 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "qbittorrent", "end": "Application"}, "context": null}
2016-12-24 11:40:22,533 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop icons", "end": "Application"}, "context": null}
2016-12-24 11:40:22,536 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:40:22,540 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run software", "end": "Application"}, "context": null}
2016-12-24 11:40:22,542 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "run", "end": "Application"}, "context": null}
2016-12-24 11:40:22,547 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "aisleriot solitaire", "end": "Application"}, "context": null}
2016-12-24 11:40:22,555 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "aisleriot", "end": "Application"}, "context": null}
2016-12-24 11:40:22,557 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "logs", "end": "Application"}, "context": null}
2016-12-24 11:40:22,561 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "swell foop", "end": "Application"}, "context": null}
2016-12-24 11:40:22,564 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "swell", "end": "Application"}, "context": null}
2016-12-24 11:40:22,567 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "taquin", "end": "Application"}, "context": null}

2016-12-24 11:40:22,570 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disks", "end": "Application"}, "context": null}
2016-12-24 11:40:22,574 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "engrampa archive manager", "end": "Application"}, "context": null}
2016-12-24 11:40:22,577 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "engrampa", "end": "Application"}, "context": null}
2016-12-24 11:40:22,579 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "i3", "end": "Application"}, "context": null}
2016-12-24 11:40:22,581 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "calculator", "end": "Application"}, "context": null}
2016-12-24 11:40:22,584 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "privilege granting", "end": "Application"}, "context": null}
2016-12-24 11:40:22,587 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "privilege", "end": "Application"}, "context": null}
2016-12-24 11:40:22,592 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tali", "end": "Application"}, "context": null}
2016-12-24 11:40:22,594 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "startup applications", "end": "Application"}, "context": null}
2016-12-24 11:40:22,599 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "startup", "end": "Application"}, "context": null}
2016-12-24 11:40:22,602 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "yaourt-gui", "end": "Application"}, "context": null}
2016-12-24 11:40:22,605 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "calendar", "end": "Application"}, "context": null}
2016-12-24 11:40:22,607 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "galculator", "end": "Application"}, "context": null}
2016-12-24 11:40:22,609 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,611 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nibbles", "end": "Application"}, "context": null}
2016-12-24 11:40:22,613 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk usage analyzer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,616 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:40:22,620 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "print preview", "end": "Application"}, "context": null}
2016-12-24 11:40:22,630 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "print", "end": "Application"}, "context": null}
2016-12-24 11:40:22,632 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth adapters", "end": "Application"}, "context": null}
2016-12-24 11:40:22,634 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:40:22,638 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system log", "end": "Application"}, "context": null}
2016-12-24 11:40:22,640 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system", "end": "Application"}, "context": null}
2016-12-24 11:40:22,644 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:40:22,647 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate disk usage analyzer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,650 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,654 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal file sharing", "end": "Application"}, "context": null}
2016-12-24 11:40:22,657 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal", "end": "Application"}, "context": null}
2016-12-24 11:40:22,660 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "date & time", "end": "Application"}, "context": null}
2016-12-24 11:40:22,662 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "date", "end": "Application"}, "context": null}
2016-12-24 11:40:22,666 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "adobe flash player", "end": "Application"}, "context": null}
2016-12-24 11:40:22,669 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "adobe", "end": "Application"}, "context": null}
2016-12-24 11:40:22,673 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote desktop viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,676 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remote", "end": "Application"}, "context": null}
2016-12-24 11:40:22,680 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice base", "end": "Application"}, "context": null}
2016-12-24 11:40:22,682 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,694 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "anjuta", "end": "Application"}, "context": null}
2016-12-24 11:40:22,698 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel preferences", "end": "Application"}, "context": null}
2016-12-24 11:40:22,700 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel", "end": "Application"}, "context": null}
2016-12-24 11:40:22,703 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gitg", "end": "Application"}, "context": null}
2016-12-24 11:40:22,706 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icc profile installer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,710 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icc", "end": "Application"}, "context": null}
2016-12-24 11:40:22,712 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pluma", "end": "Application"}, "context": null}
2016-12-24 11:40:22,714 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "vlc media player", "end": "Application"}, "context": null}

2016-12-24 11:40:22,717 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "vlc", "end": "Application"}, "context": null}
2016-12-24 11:40:22,721 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "volwheel", "end": "Application"}, "context": null}
2016-12-24 11:40:22,724 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "re\u00adgion & lan\u00adguage", "end": "Application"}, "context": null}
2016-12-24 11:40:22,727 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "re\u00adgion", "end": "Application"}, "context": null}
2016-12-24 11:40:22,729 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "Application"}, "context": null}
2016-12-24 11:40:22,732 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uxterm", "end": "Application"}, "context": null}
2016-12-24 11:40:22,737 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "help", "end": "Application"}, "context": null}
2016-12-24 11:40:22,740 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sudoku", "end": "Application"}, "context": null}
2016-12-24 11:40:22,743 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice calc", "end": "Application"}, "context": null}
2016-12-24 11:40:22,745 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:22,747 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "visual studio code", "end": "Application"}, "context": null}
2016-12-24 11:40:22,749 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "visual", "end": "Application"}, "context": null}
2016-12-24 11:40:22,751 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "character map", "end": "Application"}, "context": null}
2016-12-24 11:40:22,754 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "character", "end": "Application"}, "context": null}
2016-12-24 11:40:22,760 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "displays", "end": "Application"}, "context": null}
2016-12-24 11:40:22,762 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "five or more", "end": "Application"}, "context": null}
2016-12-24 11:40:22,765 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "five", "end": "Application"}, "context": null}
2016-12-24 11:40:22,773 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "de\u00adtails", "end": "Application"}, "context": null}
2016-12-24 11:40:22,777 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi zeroconf browser", "end": "Application"}, "context": null}
2016-12-24 11:40:22,779 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:40:22,781 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk image writer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,783 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:40:22,785 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tetravex", "end": "Application"}, "context": null}
2016-12-24 11:40:22,795 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "four-in-a-row", "end": "Application"}, "context": null}
2016-12-24 11:40:22,807 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system monitor", "end": "Application"}, "context": null}
2016-12-24 11:40:22,812 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "system", "end": "Application"}, "context": null}
2016-12-24 11:40:22,818 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "documents", "end": "Application"}, "context": null}
2016-12-24 11:40:22,828 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "videos", "end": "Application"}, "context": null}
2016-12-24 11:40:22,835 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where am i?", "end": "Application"}, "context": null}
2016-12-24 11:40:22,838 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where", "end": "Application"}, "context": null}
2016-12-24 11:40:22,841 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "screenshot", "end": "Application"}, "context": null}
2016-12-24 11:40:22,843 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "clocks", "end": "Application"}, "context": null}
2016-12-24 11:40:22,847 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "books", "end": "Application"}, "context": null}
2016-12-24 11:40:22,849 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "music", "end": "Application"}, "context": null}
2016-12-24 11:40:22,852 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lights off", "end": "Application"}, "context": null}
2016-12-24 11:40:22,855 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lights", "end": "Application"}, "context": null}
2016-12-24 11:40:22,858 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network connections", "end": "Application"}, "context": null}
2016-12-24 11:40:22,861 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:40:22,864 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound recorder", "end": "Application"}, "context": null}
2016-12-24 11:40:22,866 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sound", "end": "Application"}, "context": null}
2016-12-24 11:40:22,869 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rygel", "end": "Application"}, "context": null}
2016-12-24 11:40:22,872 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard layout", "end": "Application"}, "context": null}
2016-12-24 11:40:22,876 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "keyboard", "end": "Application"}, "context": null}
2016-12-24 11:40:22,880 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icon browser", "end": "Application"}, "context": null}

2016-12-24 11:40:22,882 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "icon", "end": "Application"}, "context": null}
2016-12-24 11:40:22,884 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rhythmbox", "end": "Application"}, "context": null}
2016-12-24 11:40:22,886 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "text editor", "end": "Application"}, "context": null}
2016-12-24 11:40:22,888 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "text", "end": "Application"}, "context": null}
2016-12-24 11:40:22,890 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gtk+ demo", "end": "Application"}, "context": null}
2016-12-24 11:40:22,891 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gtk+", "end": "Application"}, "context": null}
2016-12-24 11:40:22,893 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "slack", "end": "Application"}, "context": null}
2016-12-24 11:40:22,898 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "manage printing", "end": "Application"}, "context": null}
2016-12-24 11:40:22,902 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "manage", "end": "Application"}, "context": null}
2016-12-24 11:40:22,917 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "terminal", "end": "Application"}, "context": null}
2016-12-24 11:40:22,920 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate system monitor", "end": "Application"}, "context": null}
2016-12-24 11:40:22,951 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:22,957 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dconf editor", "end": "Application"}, "context": null}
2016-12-24 11:40:22,966 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "dconf", "end": "Application"}, "context": null}
2016-12-24 11:40:22,970 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "document viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:22,974 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "document", "end": "Application"}, "context": null}
2016-12-24 11:40:22,976 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "popup notifications", "end": "Application"}, "context": null}
2016-12-24 11:40:22,979 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "popup", "end": "Application"}, "context": null}
2016-12-24 11:40:22,983 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi ssh server browser", "end": "Application"}, "context": null}
2016-12-24 11:40:22,988 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "avahi", "end": "Application"}, "context": null}
2016-12-24 11:40:22,994 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "back\u00adground", "end": "Application"}, "context": null}
2016-12-24 11:40:22,998 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "to do", "end": "Application"}, "context": null}
2016-12-24 11:40:23,000 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "to", "end": "Application"}, "context": null}
2016-12-24 11:40:23,003 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "inkscape", "end": "Application"}, "context": null}
2016-12-24 11:40:23,006 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "photos", "end": "Application"}, "context": null}
2016-12-24 11:40:23,010 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "on\u00adline accounts", "end": "Application"}, "context": null}
2016-12-24 11:40:23,013 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "on\u00adline", "end": "Application"}, "context": null}
2016-12-24 11:40:23,016 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power management", "end": "Application"}, "context": null}
2016-12-24 11:40:23,022 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "power", "end": "Application"}, "context": null}
2016-12-24 11:40:23,025 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "po\u00adwer", "end": "Application"}, "context": null}
2016-12-24 11:40:23,028 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "main menu", "end": "Application"}, "context": null}
2016-12-24 11:40:23,034 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "main", "end": "Application"}, "context": null}
2016-12-24 11:40:23,037 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "ipython", "end": "Application"}, "context": null}
2016-12-24 11:40:23,040 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pycharm", "end": "Application"}, "context": null}
2016-12-24 11:40:23,043 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tixati", "end": "Application"}, "context": null}
2016-12-24 11:40:23,046 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network proxy", "end": "Application"}, "context": null}
2016-12-24 11:40:23,048 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:40:23,052 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uni\u00adver\u00adsal access", "end": "Application"}, "context": null}
2016-12-24 11:40:23,055 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "uni\u00adver\u00adsal", "end": "Application"}, "context": null}
2016-12-24 11:40:23,059 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake terminal", "end": "Application"}, "context": null}
2016-12-24 11:40:23,061 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "guake", "end": "Application"}, "context": null}
2016-12-24 11:40:23,065 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database access control center", "end": "Application"}, "context": null}
2016-12-24 11:40:23,068 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database", "end": "Application"}, "context": null}
2016-12-24 11:40:23,071 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atomix", "end": "Application"}, "context": null}

2016-12-24 11:40:23,088 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "color", "end": "Application"}, "context": null}
2016-12-24 11:40:23,091 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:23,094 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "font", "end": "Application"}, "context": null}
2016-12-24 11:40:23,097 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth transfer", "end": "Application"}, "context": null}
2016-12-24 11:40:23,099 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "bluetooth", "end": "Application"}, "context": null}
2016-12-24 11:40:23,102 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "access prompt", "end": "Application"}, "context": null}
2016-12-24 11:40:23,106 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "access", "end": "Application"}, "context": null}
2016-12-24 11:40:23,109 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}
2016-12-24 11:40:23,112 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "terminator", "end": "Application"}, "context": null}
2016-12-24 11:40:23,114 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "evolution", "end": "Application"}, "context": null}
2016-12-24 11:40:23,118 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "widget factory", "end": "Application"}, "context": null}
2016-12-24 11:40:23,137 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "widget", "end": "Application"}, "context": null}
2016-12-24 11:40:23,147 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network login", "end": "Application"}, "context": null}
2016-12-24 11:40:23,150 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network", "end": "Application"}, "context": null}
2016-12-24 11:40:23,157 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice impress", "end": "Application"}, "context": null}
2016-12-24 11:40:23,159 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "libreoffice", "end": "Application"}, "context": null}
2016-12-24 11:40:23,162 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "key\u00adboard", "end": "Application"}, "context": null}
2016-12-24 11:40:23,164 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "appearance", "end": "Application"}, "context": null}
2016-12-24 11:40:23,168 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop sharing", "end": "Application"}, "context": null}
2016-12-24 11:40:23,171 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "desktop", "end": "Application"}, "context": null}
2016-12-24 11:40:23,174 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "htop", "end": "Application"}, "context": null}
2016-12-24 11:40:23,179 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search and indexing", "end": "Application"}, "context": null}
2016-12-24 11:40:23,183 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "search", "end": "Application"}, "context": null}
2016-12-24 11:40:23,185 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gitkraken", "end": "Application"}, "context": null}
2016-12-24 11:40:23,189 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database browser", "end": "Application"}, "context": null}
2016-12-24 11:40:23,193 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "database", "end": "Application"}, "context": null}
2016-12-24 11:40:23,196 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "builder", "end": "Application"}, "context": null}
2016-12-24 11:40:23,199 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wa\u00adcom tab\u00adlet", "end": "Application"}, "context": null}
2016-12-24 11:40:23,204 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wa\u00adcom", "end": "Application"}, "context": null}
2016-12-24 11:40:23,207 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "devhelp", "end": "Application"}, "context": null}
2016-12-24 11:40:23,210 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk image mounter", "end": "Application"}, "context": null}
2016-12-24 11:40:23,218 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "disk", "end": "Application"}, "context": null}
2016-12-24 11:40:23,220 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "files", "end": "Application"}, "context": null}
2016-12-24 11:40:23,225 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome shell", "end": "Application"}, "context": null}
2016-12-24 11:40:23,228 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "gnome", "end": "Application"}, "context": null}
2016-12-24 11:40:23,231 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate search tool", "end": "Application"}, "context": null}
2016-12-24 11:40:23,234 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:23,237 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal file sharing", "end": "Application"}, "context": null}
2016-12-24 11:40:23,240 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "personal", "end": "Application"}, "context": null}
2016-12-24 11:40:23,242 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "caja", "end": "Application"}, "context": null}
2016-12-24 11:40:23,245 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "preferred applications", "end": "Application"}, "context": null}
2016-12-24 11:40:23,247 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "preferred", "end": "Application"}, "context": null}
2016-12-24 11:40:23,252 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "contacts", "end": "Application"}, "context": null}
2016-12-24 11:40:23,254 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "blue\u00adtooth", "end": "Application"}, "context": null}

2016-12-24 11:40:23,257 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "empathy", "end": "Application"}, "context": null}
2016-12-24 11:40:23,260 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atril document viewer", "end": "Application"}, "context": null}
2016-12-24 11:40:23,262 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atril", "end": "Application"}, "context": null}
2016-12-24 11:40:23,264 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "xterm", "end": "Application"}, "context": null}
2016-12-24 11:40:23,265 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate terminal", "end": "Application"}, "context": null}
2016-12-24 11:40:23,267 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mate", "end": "Application"}, "context": null}
2016-12-24 11:40:23,269 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "marco", "end": "Application"}, "context": null}
2016-12-24 11:40:23,274 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spotify", "end": "Application"}, "context": null}
2016-12-24 11:40:23,282 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["LaunchKeyword", "LaunchKeyword"], ["Application", "Application"]], "optional": [], "name": "LaunchDesktopApplicationIntent"}, "context": null}
2016-12-24 11:40:23,286 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["LaunchKeyword", "LaunchKeyword"], ["Website", "Website"]], "optional": [], "name": "LaunchWebsiteIntent"}, "context": null}
2016-12-24 11:40:23,289 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SearchKeyword", "SearchKeyword"], ["Website", "Website"], ["SearchTerms", "SearchTerms"]], "optional": [], "name": "SearchWebsiteIntent"}, "context": null}
2016-12-24 11:40:23,300 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "call", "end": "DialCallKeyword"}, "context": null}
2016-12-24 11:40:23,302 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "phone", "end": "DialCallKeyword"}, "context": null}
2016-12-24 11:40:23,305 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(call|phone) (?P<Contact>.*)"}, "context": null}
2016-12-24 11:40:23,310 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["DialCallKeyword", "DialCallKeyword"], ["Contact", "Contact"]], "optional": [], "name": "DialCallIntent"}, "context": null}
2016-12-24 11:40:23,313 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hello world", "end": "HelloWorldKeyword"}, "context": null}
2016-12-24 11:40:23,317 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "greetings", "end": "HelloWorldKeyword"}, "context": null}
2016-12-24 11:40:23,320 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how are you", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,322 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how have you been", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,324 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "how has your day been", "end": "HowAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,325 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thank you", "end": "ThankYouKeyword"}, "context": null}
2016-12-24 11:40:23,327 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "thanks", "end": "ThankYouKeyword"}, "context": null}
2016-12-24 11:40:23,329 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ThankYouKeyword", "ThankYouKeyword"]], "optional": [], "name": "ThankYouIntent"}, "context": null}
2016-12-24 11:40:23,331 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["HowAreYouKeyword", "HowAreYouKeyword"]], "optional": [], "name": "HowAreYouIntent"}, "context": null}
2016-12-24 11:40:23,333 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["HelloWorldKeyword", "HelloWorldKeyword"]], "optional": [], "name": "HelloWorldIntent"}, "context": null}
2016-12-24 11:40:23,345 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "IP address", "end": "IPCommand"}, "context": null}
2016-12-24 11:40:23,347 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "IP", "end": "IPCommand"}, "context": null}
2016-12-24 11:40:23,351 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "network address", "end": "IPCommand"}, "context": null}
2016-12-24 11:40:23,354 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["IPCommand", "IPCommand"]], "optional": [], "name": "IPIntent"}, "context": null}
2016-12-24 11:40:23,357 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "joke", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:40:23,360 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "make me laugh", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:40:23,363 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "brighten my day", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:40:23,368 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me a joke", "end": "JokingKeyword"}, "context": null}
2016-12-24 11:40:23,374 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["JokingKeyword", "JokingKeyword"]], "optional": [], "name": "JokingIntent"}, "context": null}
2016-12-24 11:40:23,378 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "go to sleep", "end": "SleepCommand"}, "context": null}
2016-12-24 11:40:23,382 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "nap time", "end": "SleepCommand"}, "context": null}
2016-12-24 11:40:23,388 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleepy time tea", "end": "SleepCommand"}, "context": null}
2016-12-24 11:40:23,393 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleepytime tea", "end": "SleepCommand"}, "context": null}
2016-12-24 11:40:23,398 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SleepCommand", "SleepCommand"]], "optional": [], "name": "NapTimeIntent"}, "context": null}

2016-12-24 11:40:23,409 - Skills - DEBUG - {"type": "enclosure.system.reset", "data": {}, "context": null}
2016-12-24 11:40:23,411 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "news", "end": "NPRNewsKeyword"}, "context": null}
2016-12-24 11:40:23,412 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me the news", "end": "NPRNewsKeyword"}, "context": null}
2016-12-24 11:40:23,417 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["NPRNewsKeyword", "NPRNewsKeyword"]], "optional": [], "name": "NPRNewsIntent"}, "context": null}
2016-12-24 11:40:23,419 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:23,422 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "temperature", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:23,425 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "forecast", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:23,430 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rain", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,433 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raining", "end": "WeatherType", "alias_of": "rain"}, "context": null}
2016-12-24 11:40:23,436 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snow", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,439 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snowing", "end": "WeatherType", "alias_of": "snow"}, "context": null}
2016-12-24 11:40:23,442 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleet", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,444 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hail", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,448 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hailing", "end": "WeatherType", "alias_of": "hail"}, "context": null}
2016-12-24 11:40:23,451 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sunny", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,456 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sun", "end": "WeatherType", "alias_of": "sunny"}, "context": null}
2016-12-24 11:40:23,458 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hot", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,460 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "warm", "end": "WeatherType", "alias_of": "hot"}, "context": null}
2016-12-24 11:40:23,462 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cold", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,464 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cool", "end": "WeatherType", "alias_of": "cold"}, "context": null}
2016-12-24 11:40:23,466 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:23,470 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tomorrow", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,475 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,477 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in 1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,480 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,483 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in one day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,486 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,488 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,491 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "following day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,494 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:23,497 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Seattle", "end": "Location"}, "context": null}
2016-12-24 11:40:23,500 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Los Angeles california", "end": "Location"}, "context": null}
2016-12-24 11:40:23,504 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "los angeles", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:40:23,507 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "la", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:40:23,510 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Lawrence kansas", "end": "Location"}, "context": null}
2016-12-24 11:40:23,514 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland oregon", "end": "Location"}, "context": null}
2016-12-24 11:40:23,519 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland", "end": "Location", "alias_of": "portland oregon"}, "context": null}
2016-12-24 11:40:23,522 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spokane", "end": "Location"}, "context": null}
2016-12-24 11:40:23,526 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "new york", "end": "Location"}, "context": null}
2016-12-24 11:40:23,530 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chicago", "end": "Location"}, "context": null}
2016-12-24 11:40:23,532 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "houston", "end": "Location"}, "context": null}
2016-12-24 11:40:23,534 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "austin", "end": "Location"}, "context": null}
2016-12-24 11:40:23,538 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san diego", "end": "Location"}, "context": null}

2016-12-24 11:40:23,540 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san francisco", "end": "Location"}, "context": null}
2016-12-24 11:40:23,542 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san jose", "end": "Location"}, "context": null}
2016-12-24 11:40:23,553 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boston", "end": "Location"}, "context": null}
2016-12-24 11:40:23,556 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "washington d c", "end": "Location"}, "context": null}
2016-12-24 11:40:23,559 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atlanta", "end": "Location"}, "context": null}
2016-12-24 11:40:23,562 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,565 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,570 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,574 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,576 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,580 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,585 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,588 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,591 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,600 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,602 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,605 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,608 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,613 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:23,617 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:40:23,619 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pair", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:40:23,622 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "pairing", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:40:23,624 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "register", "end": "PairingKeyword"}, "context": null}
2016-12-24 11:40:23,627 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "unit", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:40:23,629 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "device", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:40:23,631 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mycroft", "end": "DeviceKeyword"}, "context": null}
2016-12-24 11:40:23,634 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["PairingKeyword", "PairingKeyword"], ["DeviceKeyword", "DeviceKeyword"]], "optional": [], "name": "PairingIntent"}, "context": null}
2016-12-24 11:40:23,636 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where were you born", "end": "WhereWereYouBornKeyword"}, "context": null}
2016-12-24 11:40:23,648 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "where were you created", "end": "WhereWereYouBornKeyword"}, "context": null}
2016-12-24 11:40:23,652 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "when were you born", "end": "WhenWereYouBornKeyword"}, "context": null}
2016-12-24 11:40:23,654 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "when were you created", "end": "WhenWereYouBornKeyword"}, "context": null}
2016-12-24 11:40:23,656 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who made you", "end": "WhoMadeYouKeyword"}, "context": null}
2016-12-24 11:40:23,658 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who were you made by", "end": "WhoMadeYouKeyword"}, "context": null}
2016-12-24 11:40:23,661 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what are you", "end": "WhatAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,667 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who are you", "end": "WhoAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,674 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "who're you", "end": "WhoAreYouKeyword"}, "context": null}
2016-12-24 11:40:23,678 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhenWereYouBornKeyword", "WhenWereYouBornKeyword"]], "optional": [], "name": "WhenWereYouBornIntent"}, "context": null}
2016-12-24 11:40:23,681 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhereWereYouBornKeyword", "WhereWereYouBornKeyword"]], "optional": [], "name": "WhereWereYouBornIntent"}, "context": null}
2016-12-24 11:40:23,684 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhoMadeYouKeyWord", "WhoMadeYouKeyWord"]], "optional": [], "name": "WhoMadeYouIntent"}, "context": null}
2016-12-24 11:40:23,687 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WhoAreYouKeyword", "WhoAreYouKeyword"]], "optional": [], "name": "WhoAreYouIntent"}, "context": null}
2016-12-24 11:40:23,690 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires":

[["WhatAreYouKeyword", "WhatAreYouKeyword"]], "optional": [], "name": "WhatAreYouIntent"}, "context": null}
2016-12-24 11:40:23,692 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:40:23,695 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "all my", "end": "ReminderSkillAmount", "alias_of": "all"}, "context": null}
2016-12-24 11:40:23,698 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:40:23,702 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "ReminderSkillAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:40:23,704 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:40:23,707 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "ReminderSkillAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:40:23,710 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:40:23,714 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following", "end": "ReminderSkillAmount"}, "context": null}
2016-12-24 11:40:23,718 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "show", "end": "ReminderSkillListVerb"}, "context": null}
2016-12-24 11:40:23,723 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "list", "end": "ReminderSkillListVerb"}, "context": null}
2016-12-24 11:40:23,726 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "erase", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:23,729 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cancel", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:23,740 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "delete", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:23,743 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remove", "end": "ReminderSkillDeleteVerb"}, "context": null}
2016-12-24 11:40:23,746 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminder", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:40:23,748 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminders", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:40:23,752 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notification", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:40:23,754 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notifications", "end": "ReminderSkillKeyword"}, "context": null}
2016-12-24 11:40:23,755 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "off", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:40:23,760 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "end", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:40:23,762 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "ReminderSkillStopVerb"}, "context": null}
2016-12-24 11:40:23,765 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remind", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,767 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notify", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,770 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "notify me", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,773 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "remind me", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,778 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reminder", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,780 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "set a reminder", "end": "ReminderSkillCreateVerb"}, "context": null}
2016-12-24 11:40:23,782 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<ReminderSkillAmount>\\d+)"}, "context": null}
2016-12-24 11:40:23,785 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillCreateVerb", "ReminderSkillCreateVerb"]], "optional": [], "name": "ReminderSkillCreateIntent"}, "context": null}
2016-12-24 11:40:23,787 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillListVerb", "ReminderSkillListVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [["ReminderSkillAmount", "ReminderSkillAmount"]], "name": "ReminderSkillListIntent"}, "context": null}
2016-12-24 11:40:23,790 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillDeleteVerb", "ReminderSkillDeleteVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [["ReminderSkillAmount", "ReminderSkillAmount"]], "name": "ReminderSkillDeleteIntent"}, "context": null}
2016-12-24 11:40:23,793 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ReminderSkillStopVerb", "ReminderSkillStopVerb"], ["ReminderSkillKeyword", "ReminderSkillKeyword"]], "optional": [], "name": "ReminderSkillStopIntent"}, "context": null}
2016-12-24 11:40:23,795 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "speak", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:40:23,797 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "say", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:40:23,799 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "repeat", "end": "SpeakKeyword"}, "context": null}
2016-12-24 11:40:23,811 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "speak (?P<Words>.*)"}, "context": null}
2016-12-24 11:40:23,814 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "say (?P<Words>.*)"}, "context": null}
2016-12-24 11:40:23,817 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "repeat (?P<Words>.*)"}, "context": null}
2016-12-24 11:40:23,825 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires":

[["SpeakKeyword", "SpeakKeyword"], ["Words", "Words"]], "optional": [], "name": "SpeakIntent"}, "context": null}
2016-12-24 11:40:23,831 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spell", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:40:23,834 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spell the word", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:40:23,837 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spelling of", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:40:23,841 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spelling of the word", "end": "SpellingKeyword"}, "context": null}
2016-12-24 11:40:23,844 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(spell the word|spelling of the word|spelling of|spell) (?P<Word>\\w+)"}, "context": null}
2016-12-24 11:40:23,847 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["SpellingKeyword", "SpellingKeyword"], ["Word", "Word"]], "optional": [], "name": "SpellingIntent"}, "context": null}
2016-12-24 11:40:23,850 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stock price", "end": "StockPriceKeyword"}, "context": null}
2016-12-24 11:40:23,853 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "trading at", "end": "StockPriceKeyword"}, "context": null}
2016-12-24 11:40:23,861 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "price of (?P<Company>.*)"}, "context": null}
2016-12-24 11:40:23,863 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "is (?P<Company>.*) trading at"}, "context": null}
2016-12-24 11:40:23,865 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["StockPriceKeyword", "StockPriceKeyword"], ["Company", "Company"]], "optional": [], "name": "StockPriceIntent"}, "context": null}
2016-12-24 11:40:23,868 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "stop", "end": "StopKeyword"}, "context": null}
2016-12-24 11:40:23,870 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "silence", "end": "StopKeyword"}, "context": null}
2016-12-24 11:40:23,872 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "shut up", "end": "StopKeyword"}, "context": null}
2016-12-24 11:40:23,876 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "be quiet", "end": "StopKeyword"}, "context": null}
2016-12-24 11:40:23,881 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["StopKeyword", "StopKeyword"]], "optional": [], "name": "StopIntent"}, "context": null}
2016-12-24 11:40:23,883 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reset volume", "end": "ResetVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,886 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "restart volume", "end": "ResetVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,890 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "lower volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,894 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "reduce volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,901 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "decrease volume", "end": "DecreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,905 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "0", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,908 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "zero", "end": "VolumeAmount", "alias_of": "0"}, "context": null}
2016-12-24 11:40:23,910 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,913 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one", "end": "VolumeAmount", "alias_of": "1"}, "context": null}
2016-12-24 11:40:23,915 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "2", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,919 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "two", "end": "VolumeAmount", "alias_of": "2"}, "context": null}
2016-12-24 11:40:23,921 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "quiet", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,925 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "normal", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,927 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "loud", "end": "VolumeAmount"}, "context": null}
2016-12-24 11:40:23,946 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "mute volume", "end": "MuteVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,953 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "volume", "end": "VolumeKeyword"}, "context": null}
2016-12-24 11:40:23,957 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rise volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,965 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raise volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,971 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boost volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,976 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "increase volume", "end": "IncreaseVolumeKeyword"}, "context": null}
2016-12-24 11:40:23,978 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(?P<VolumeAmount>\\d+)"}, "context": null}
2016-12-24 11:40:24,002 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["VolumeKeyword", "VolumeKeyword"], ["VolumeAmount", "VolumeAmount"]], "optional": [], "name": "SetVolumeIntent"}, "context": null}
2016-12-24 11:40:24,005 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["IncreaseVolumeKeyword", "IncreaseVolumeKeyword"]], "optional": [], "name": "IncreaseVolumeIntent"}, "context": null}
2016-12-24 11:40:24,010 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires":

[["DecreaseVolumeKeyword", "DecreaseVolumeKeyword"]], "optional": [], "name": "DecreaseVolumeIntent"}, "context": null}
2016-12-24 11:40:24,013 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["MuteVolumeKeyword", "MuteVolumeKeyword"]], "optional": [], "name": "MuteVolumeIntent"}, "context": null}
2016-12-24 11:40:24,020 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["ResetVolumeKeyword", "ResetVolumeKeyword"]], "optional": [], "name": "ResetVolumeIntent"}, "context": null}
2016-12-24 11:40:24,027 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "weather", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:24,040 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "temperature", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:24,045 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "forecast", "end": "WeatherKeyword"}, "context": null}
2016-12-24 11:40:24,047 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "rain", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,066 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "raining", "end": "WeatherType", "alias_of": "rain"}, "context": null}
2016-12-24 11:40:24,075 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snow", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,077 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "snowing", "end": "WeatherType", "alias_of": "snow"}, "context": null}
2016-12-24 11:40:24,080 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sleet", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,084 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hail", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,091 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hailing", "end": "WeatherType", "alias_of": "hail"}, "context": null}
2016-12-24 11:40:24,101 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sunny", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,104 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "sun", "end": "WeatherType", "alias_of": "sunny"}, "context": null}
2016-12-24 11:40:24,115 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "hot", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,118 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "warm", "end": "WeatherType", "alias_of": "hot"}, "context": null}
2016-12-24 11:40:24,122 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cold", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,125 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "cool", "end": "WeatherType", "alias_of": "cold"}, "context": null}
2016-12-24 11:40:24,130 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "", "end": "WeatherType"}, "context": null}
2016-12-24 11:40:24,132 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tomorrow", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,135 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,137 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in 1 day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,143 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "one day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,147 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in one day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,159 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,165 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,167 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "following day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,171 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the following day", "end": "NextDay"}, "context": null}
2016-12-24 11:40:24,180 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Seattle", "end": "Location"}, "context": null}
2016-12-24 11:40:24,190 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Los Angeles california", "end": "Location"}, "context": null}
2016-12-24 11:40:24,197 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "los angeles", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:40:24,200 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "la", "end": "Location", "alias_of": "Los Angeles california"}, "context": null}
2016-12-24 11:40:24,204 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "Lawrence kansas", "end": "Location"}, "context": null}
2016-12-24 11:40:24,213 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland oregon", "end": "Location"}, "context": null}
2016-12-24 11:40:24,220 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "portland", "end": "Location", "alias_of": "portland oregon"}, "context": null}
2016-12-24 11:40:24,225 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "spokane", "end": "Location"}, "context": null}
2016-12-24 11:40:24,230 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "new york", "end": "Location"}, "context": null}
2016-12-24 11:40:24,241 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "chicago", "end": "Location"}, "context": null}
2016-12-24 11:40:24,256 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "houston", "end": "Location"}, "context": null}
2016-12-24 11:40:24,259 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "austin", "end": "Location"}, "context": null}

2016-12-24 11:40:24,263 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san diego", "end": "Location"}, "context": null}
2016-12-24 11:40:24,273 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san francisco", "end": "Location"}, "context": null}
2016-12-24 11:40:24,276 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "san jose", "end": "Location"}, "context": null}
2016-12-24 11:40:24,278 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "boston", "end": "Location"}, "context": null}
2016-12-24 11:40:24,296 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "washington d c", "end": "Location"}, "context": null}
2016-12-24 11:40:24,301 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "atlanta", "end": "Location"}, "context": null}
2016-12-24 11:40:24,311 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,316 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,319 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hour", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,325 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,331 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,334 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,338 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,346 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,349 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,355 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next few hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,363 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,365 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,371 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,380 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "in the next couple of hours", "end": "NextHours"}, "context": null}
2016-12-24 11:40:24,386 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(at|in) (?P<Location>.*)"}, "context": null}
2016-12-24 11:40:24,390 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"]], "optional": [["Location", "Location"]], "name": "CurrentWeatherIntent"}, "context": null}
2016-12-24 11:40:24,394 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"], ["NextHours", "NextHours"]], "optional": [["Location", "Location"]], "name": "NextHoursWeatherIntent"}, "context": null}
2016-12-24 11:40:24,400 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WeatherKeyword", "WeatherKeyword"], ["NextDay", "NextDay"]], "optional": [["Location", "Location"]], "name": "NextDayWeatherIntent"}, "context": null}
2016-12-24 11:40:24,407 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wiki", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:40:24,420 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "wikipedia", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:40:24,425 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell me about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:40:24,436 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "tell us about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:40:24,440 - Skills - DEBUG - {"type": "register_vocab", "data": {"start": "what does wikipedia say about", "end": "WikipediaKeyword"}, "context": null}
2016-12-24 11:40:24,454 - Skills - DEBUG - {"type": "register_vocab", "data": {"regex": "(tell us about|tell me about|what does wikipedia say about|wiki|wikipedia) (?P<ArticleTitle>.*)"}, "context": null}
2016-12-24 11:40:24,460 - Skills - DEBUG - {"type": "register_intent", "data": {"at_least_one": [], "requires": [["WikipediaKeyword", "WikipediaKeyword"], ["ArticleTitle", "ArticleTitle"]], "optional": [], "name": "WikipediaIntent"}, "context": null}
2016-12-24 11:40:36,459 - Skills - DEBUG - {"type": "recognizer_loop:utterance", "data": {"utterances": ["tell me about my chemical romance"]}, "context": null}
2016-12-24 11:40:36,513 - Skills - DEBUG - {"type": "WikipediaIntent", "data": {"confidence": 0.6818181818181819, "target": null, "WikipediaKeyword": "tell me about", "intent_type": "WikipediaIntent", "ArticleTitle": "my chemical romance", "utterance": "tell me about my chemical romance"}, "context": {"target": null}}
2016-12-24 11:40:36,534 - Skills - DEBUG - {"type": "speak", "data": {"utterance": "Okay, I am searching for my chemical romance"}, "context": null}
2016-12-24 11:40:36,542 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTP connection (1): en.wikipedia.org
2016-12-24 11:40:36,563 - Skills - DEBUG - {"type": "recognizer_loop:audio_output_start", "data": {}, "context": null}
2016-12-24 11:40:36,954 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?format=json&srsearch=my+chemical+romance&list=search&srlimit=5&srprop=&limit=5&action=query HTTP/1.1" 301 0
2016-12-24 11:40:36,968 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): en.wikipedia.org
2016-12-24 11:40:37,037 - Skills - DEBUG - {"type": "enclosure.eyes.blink", "data": {"side": "b"}, "context": null}

2016-12-24 11:40:37,076 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:37,314 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:37,404 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:37,501 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:37,749 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:37,953 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:38,064 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?format=json&srsearch=my+chemical+romance&list=search&srlimit=5&srprop=&limit=5&action=query HTTP/1.1" 200 248
2016-12-24 11:40:38,089 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:38,092 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTP connection (1): en.wikipedia.org
2016-12-24 11:40:38,157 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:38,225 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:38,323 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:38,412 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:38,497 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?srinfo=suggestion&srsearch=My+Chemical+Romance&format=json&srlimit=1&list=search&srprop=&limit=1&action=query HTTP/1.1" 301 0
2016-12-24 11:40:38,511 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): en.wikipedia.org
2016-12-24 11:40:38,544 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:38,610 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:38,740 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "5"}, "context": null}
2016-12-24 11:40:38,867 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "6"}, "context": null}
2016-12-24 11:40:38,928 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:38,944 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:39,020 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:39,124 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:39,231 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:39,307 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:39,364 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:39,420 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:39,501 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:39,554 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:39,622 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:39,654 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:39,665 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?srinfo=suggestion&srsearch=My+Chemical+Romance&format=json&srlimit=1&list=search&srprop=&limit=1&action=query HTTP/1.1" 200 160
2016-12-24 11:40:39,687 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTP connection (1): en.wikipedia.org
2016-12-24 11:40:39,718 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:39,822 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:39,877 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:39,959 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:40,070 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:40,165 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?inprop=url&redirects=&format=json&ppprop=disambiguation&prop=info%7Cpageprops&titles=My+Chemical+Romance&action=query HTTP/1.1" 301 0
2016-12-24 11:40:40,178 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): en.wikipedia.org

2016-12-24 11:40:40,267 - Skills - DEBUG - {"type": "recognizer_loop:audio_output_end", "data": {}, "context": null}
2016-12-24 11:40:41,344 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?inprop=url&redirects=&format=json&ppprop=disambiguation&prop=info%7Cpageprops&titles=My+Chemical+Romance&action=query HTTP/1.1" 200 280
2016-12-24 11:40:41,364 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTP connection (1): en.wikipedia.org
2016-12-24 11:40:41,780 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?format=json&exsentences=2&prop=extracts&titles=My+Chemical+Romance&action=query&explaintext= HTTP/1.1" 301 0
2016-12-24 11:40:41,794 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): en.wikipedia.org
2016-12-24 11:40:42,860 - requests.packages.urllib3.connectionpool - DEBUG - "GET /w/api.php?format=json&exsentences=2&prop=extracts&titles=My+Chemical+Romance&action=query&explaintext= HTTP/1.1" 200 277
2016-12-24 11:40:42,878 - Skills - DEBUG - {"type": "speak", "data": {"utterance": "My Chemical Romance was an American rock band from New Jersey, active from 2001 to 2013. The band's best-known lineup consisted of lead vocalist Gerard Way, guitarists Ray Toro and Frank Iero, bassist Mikey Way and drummer Bob Bryar."}, "context": null}
2016-12-24 11:40:42,885 - Skills - DEBUG - {"type": "recognizer_loop:audio_output_start", "data": {}, "context": null}
2016-12-24 11:40:43,812 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:44,035 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:44,072 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:44,172 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:44,232 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:44,301 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:44,412 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:44,450 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:44,550 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:44,604 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:44,643 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:44,691 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:44,740 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:44,800 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:44,881 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:44,932 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:45,008 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:45,065 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,090 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:45,178 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,235 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:45,258 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,314 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:45,380 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,459 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:45,562 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,565 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:45,664 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,713 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:45,777 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:45,831 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:45,881 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}

2016-12-24 11:40:46,015 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:46,044 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:46,119 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:46,181 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:46,261 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "5"}, "context": null}
2016-12-24 11:40:46,351 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:46,398 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:46,409 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:46,476 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:46,501 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:46,590 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:46,737 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:46,925 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:47,036 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:47,247 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:47,459 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:47,542 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:47,603 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:47,653 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:47,684 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "5"}, "context": null}
2016-12-24 11:40:47,738 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "5"}, "context": null}
2016-12-24 11:40:47,821 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:47,881 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:47,903 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "4"}, "context": null}
2016-12-24 11:40:47,964 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,064 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:48,146 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,263 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "1"}, "context": null}
2016-12-24 11:40:48,409 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,514 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:48,561 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,640 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,645 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:48,734 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:48,771 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,833 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:48,920 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
2016-12-24 11:40:48,966 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "3"}, "context": null}
2016-12-24 11:40:49,082 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "2"}, "context": null}
2016-12-24 11:40:49,123 - Skills - DEBUG - {"type": "enclosure.mouth.viseme", "data": {"code": "0"}, "context": null}
/home/paul/Irene/init/start.sh: line 26: 29476 Terminated PYTHONPATH=${TOP} python ${SCRIPT} $@

Directory: intelora-core/logs/intelora-voice.log

Starting voice
2016-12-24 11:38:27,963 - mycroft.configuration - DEBUG - Configuration '/home/paul/Irene/mycroft/configuration/mycroft.conf' loaded

2016-12-24 11:38:27,964 - mycroft.configuration - DEBUG - Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:38:27,965 - mycroft.configuration - DEBUG - Configuration '/home/paul/.mycroft/mycroft.conf' not found
2016-12-24 11:38:28,265 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:38:29,542 - requests.packages.urllib3.connectionpool - DEBUG - "GET /v1/device/c14b1936-4458-4859-9cab-ebb56da21994/setting HTTP/1.1" 200 2887
Carnegie Mellon University, Copyright (c) 1999-2011, all rights reserved
version: mimic-2.0.0-release Dec 2014 (http://cmuflite.org)
ALSA lib pcm_dsnoop.c:618:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1041:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1041:(snd_pcm_dmix_open) unable to open slave
connect(2) call to /dev/shm/jack-1000/default/jack_0 failed (err=No such file or directory)
attempt to connect to server failed
2016-12-24 11:38:31,411 - mycroft.messagebus.client.ws - INFO - Connected
2016-12-24 11:38:32,391 - mycroft.client.speech.mic - DEBUG - Waiting for wake word...
2016-12-24 11:38:42,338 - SpeechClient - INFO - Speak: Tim (movie) : title, Tim, director, Michael Pate, release date, September 17, 1981 (35 years 3 months ago), runtime, 109 minutes (1 hour 49 minutes), writers, Colleen McCullough, Michael Pate, genres, drama, romance
Playing WAVE '/tmp/tts.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
/home/paul/Irene/init/start.sh: line 26: 28995 Terminated PYTHONPATH=${TOP} python ${SCRIPT} $@
Starting voice
2016-12-24 11:40:17,030 - mycroft.configuration - DEBUG - Configuration '/home/paul/Irene/mycroft/configuration/mycroft.conf' loaded
2016-12-24 11:40:17,031 - mycroft.configuration - DEBUG - Configuration '/etc/mycroft/mycroft.conf' not found
2016-12-24 11:40:17,031 - mycroft.configuration - DEBUG - Configuration '/home/paul/.mycroft/mycroft.conf' not found
2016-12-24 11:40:17,323 - requests.packages.urllib3.connectionpool - INFO - Starting new HTTPS connection (1): api.mycroft.ai
2016-12-24 11:40:18,677 - requests.packages.urllib3.connectionpool - DEBUG - "GET /v1/device/c14b1936-4458-4859-9cab-ebb56da21994/setting HTTP/1.1" 200 2887
Carnegie Mellon University, Copyright (c) 1999-2011, all rights reserved
version: mimic-2.0.0-release Dec 2014 (http://cmuflite.org)
ALSA lib pcm_dsnoop.c:618:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1041:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2450:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1041:(snd_pcm_dmix_open) unable to open slave
connect(2) call to /dev/shm/jack-1000/default/jack_0 failed (err=No such file or directory)
attempt to connect to server failed
2016-12-24 11:40:20,597 - mycroft.messagebus.client.ws - INFO - Connected
2016-12-24 11:40:21,599 - mycroft.client.speech.mic - DEBUG - Waiting for wake word...
2016-12-24 11:40:36,524 - SpeechClient - INFO - Speak: Okay, I am searching for my chemical romance
Playing WAVE '/tmp/tts.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
2016-12-24 11:40:42,880 - SpeechClient - INFO - Speak: My Chemical Romance was an American rock band from New Jersey, active from 2001 to 2013.
Playing WAVE '/tmp/tts.wav' : Signed 16 bit Little Endian, Rate 16000 Hz, Mono
/home/paul/Irene/init/start.sh: line 26: 29491 Terminated PYTHONPATH=${TOP} python ${SCRIPT} $@

Directory: intelora-core/misc/backup/google_tts (forslund).py

```
# Copyright 2016 Mycroft AI, Inc.
#
# This file is part of Mycroft Core.
#
# Mycroft Core is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
```

```
# Mycroft Core is distributed in the hope that it will be
useful,
# but WITHOUT ANY WARRANTY; without even the implied
warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General
Public License
# along with Mycroft Core.  If not, see
<http://www.gnu.org/licenses/>.

from gtts import gTTS

from mycroft.tts import TTS, TTSValidator
from mycroft.util import play_mp3

__author__ = 'jdorleans'

NAME = 'gtts'


class GoogleTTS(TTS):
    def __init__(self, lang, voice):
        super(GoogleTTS, self).__init__(lang, voice)

    def execute(self, sentence, client):
        for s in sentence.split('.'):
            tts = gTTS(text=s)
            tts.save('/tmp/gtts.mp3')
            play_mp3('/tmp/gtts.mp3')

class GoogleTTSValidator(TTSValidator):
    def __init__(self):
        super(GoogleTTSValidator, self).__init__()

    def validate_lang(self, lang):
        # TODO
        pass

    def validate_connection(self, tts):
        try:
            gTTS(text='Hi').save(tts.filename)
        except:
            raise Exception(
                'GoogleTTS server could not be verified. Please
check your '
                'internet connection.')

    def get_instance(self):
        return GoogleTTS
```

Directory: intelora-core/misc/backup/google_tts (jcasoft).py

```
# Copyright 2016 Mycroft AI, Inc.
#
# This file is part of Mycroft Core.
#
# Mycroft Core is free software: you can redistribute it
and/or modify
# it under the terms of the GNU General Public License as
published by
# the Free Software Foundation, either version 3 of the
License, or
# (at your option) any later version.
#
# Mycroft Core is distributed in the hope that it will be
useful,
# but WITHOUT ANY WARRANTY; without even the implied
warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General
Public License
# along with Mycroft Core.  If not, see
<http://www.gnu.org/licenses/>.

from mycroft.configuration import ConfigurationManager
from mycroft.tts import TTS, TTSValidator
from mycroft.util import play_mp3
import os

__author__ = 'jcasoft'
core_config = ConfigurationManager.get().get('core')

NAME = 'gtts'

class GoogleTTS(TTS):
    def __init__(self, lang, voice):
        super(GoogleTTS, self).__init__(lang, voice)
            self.language = core_config.get('lang')

    def execute(self, sentence, client):
            lang = self.language
            get_sentence = 'wget -q -U Mozilla -O
/tmp/gtts.mp3
"https://translate.google.com/translate_tts?tl=' + lang +
'&q='+ sentence +'&client=tw-ob'+'"'
            os.system(get_sentence)
            play_mp3("/tmp/gtts.mp3")

class GoogleTTSValidator(TTSValidator):
    def __init__(self):
        super(GoogleTTSValidator, self).__init__()

    def validate_lang(self, lang):
        # TODO
        pass

    def validate_connection(self, tts):
        try:
                pass
        except:
            raise Exception(
                'GTTS not connected. Please check your internet
connection.')

    def get_instance(self):
        return GoogleTTS
```

Directory: intelora-core/misc/backup/google_tts (original).py

```
# Copyright 2016 Mycroft AI, Inc.
#
```

```
# This file is part of Mycroft Core.
#
# Mycroft Core is free software: you can redistribute it
and/or modify
# it under the terms of the GNU General Public License as
published by
# the Free Software Foundation, either version 3 of the
License, or
# (at your option) any later version.
#
# Mycroft Core is distributed in the hope that it will be
useful,
# but WITHOUT ANY WARRANTY; without even the implied
warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General
Public License
# along with Mycroft Core.  If not, see
<http://www.gnu.org/licenses/>.

from gtts import gTTS

from mycroft.tts import TTS, TTSValidator
from mycroft.util import play_wav

__author__ = 'jdorleans'

NAME = 'gtts'


class GoogleTTS(TTS):
    def __init__(self, lang, voice):
        super(GoogleTTS, self).__init__(lang, voice)

    def execute(self, sentence, client):
        tts = gTTS(text=sentence, lang=self.lang)
        tts.save(self.filename)
        play_wav(self.filename)


class GoogleTTSValidator(TTSValidator):
    def __init__(self):
        super(GoogleTTSValidator, self).__init__()

    def validate_lang(self, lang):
        # TODO
        pass

    def validate_connection(self, tts):
        try:
            gTTS(text='Hi').save(tts.filename)
        except:
            raise Exception(
                'GoogleTTS server could not be verified. Please
check your '
                'internet connection.')

    def get_instance(self):
        return GoogleTTS
```

Directory: intelora-core/mycroft/api/__init__.py

```
import requests
from requests import HTTPError

from mycroft.configuration import ConfigurationManager
from mycroft.identity import IdentityManager
from mycroft.version import VersionManager

__author__ = 'jdorleans'


class Api(object):
    def __init__(self, path):
        self.path = path
        config = ConfigurationManager().get()
        config_server = config.get("server")
        self.url = config_server.get("url")
        self.version = config_server.get("version")
        self.identity = IdentityManager.get()

    def check_token(self):
        if self.identity.refresh and self.identity.is_expired():
            self.identity = IdentityManager.load()
            if self.identity.is_expired():
                data = self.send({
                    "path": "auth/token",
                    "headers": {
                        "Authorization": "Bearer " +
self.identity.refresh
                    }
                })
                IdentityManager.save(data)

    def send(self, params):
        method = params.get("method", "GET")
        headers = self.build_headers(params)
        data = self.build_data(params)
        json = self.build_json(params)
        query = self.build_query(params)
        url = self.build_url(params)
        response = requests.request(method, url,
headers=headers, params=query,
                                    data=data, json=json)
        return self.get_response(response)

    def request(self, params):
        self.check_token()
        self.build_path(params)
        return self.send(params)

    def get_response(self, response):
        data = self.get_data(response)
        if 200 <= response.status_code < 300:
            return data
        raise HTTPError(data, response=response)

    def get_data(self, response):
        try:
            return response.json()
        except:
            return response.text

    def build_headers(self, params):
```

```
        headers = params.get("headers", {})
        self.add_content_type(headers)
        self.add_authorization(headers)
        params["headers"] = headers
        return headers

    def add_content_type(self, headers):
        if not headers.__contains__("Content-Type"):
            headers["Content-Type"] = "application/json"

    def add_authorization(self, headers):
        if not headers.__contains__("Authorization"):
            headers["Authorization"] = "Bearer " +
self.identity.access

    def build_data(self, params):
        return params.get("data")

    def build_json(self, params):
        json = params.get("json")
        if json and params["headers"]["Content-Type"] ==
"application/json":
            for k, v in json.iteritems():
                if v == "":
                    json[k] = None
            params["json"] = json
        return json

    def build_query(self, params):
        return params.get("query")

    def build_path(self, params):
        path = params.get("path", "")
        params["path"] = self.path + path
        return params["path"]

    def build_url(self, params):
        path = params.get("path", "")
        version = params.get("version", self.version)
        return self.url + "/" + version + "/" + path


class DeviceApi(Api):
    def __init__(self):
        super(DeviceApi, self).__init__("device")

    def get_code(self, state):
        IdentityManager.update()
        return self.request({
            "path": "/code?state=" + state
        })

    def activate(self, state, token):
        version = VersionManager.get()
        return self.request({
            "method": "POST",
            "path": "/activate",
            "json": {"state": state,
                     "token": token,
                     "coreVersion": version.get("coreVersion"),
                     "enclosureVersion":
version.get("enclosureVersion")}
        })

    def find(self):
        return self.request({
            "path": "/" + self.identity.uuid
        })

    def find_setting(self):
        return self.request({
            "path": "/" + self.identity.uuid + "/setting"
        })


class STTApi(Api):
    def __init__(self):
        super(STTApi, self).__init__("stt")

    def stt(self, audio, language, limit):
        return self.request({
            "method": "POST",
            "headers": {"Content-Type": "audio/x-flac"},
            "query": {"lang": language, "limit": limit},
            "data": audio
        })
```

Directory: intelora-core/mycroft/client/enclore/__init__.py

```
import subprocess
import time
from Queue import Queue
from alsaaudio import Mixer
from threading import Thread, Timer

import serial

from mycroft.client.enclosure.arduino import
EnclosureArduino
from mycroft.client.enclosure.eyes import EnclosureEyes
from mycroft.client.enclosure.mouth import
EnclosureMouth
from mycroft.client.enclosure.weather import
EnclosureWeather
from mycroft.configuration import ConfigurationManager
from mycroft.messagebus.client.ws import WebsocketClient
from mycroft.messagebus.message import Message
from mycroft.util import play_wav, create_signal
from mycroft.util.audio_test import record
from mycroft.util.log import getLogger

__author__ = 'aatchison', 'jdorleans', 'iward'

LOG = getLogger("EnclosureClient")


class EnclosureReader(Thread):
    """
    Reads data from Serial port.

    Listens to all commands sent by Arduino that must be be
performed on
    Mycroft Core.

    E.g. Mycroft Stop Feature
        #. Arduino sends a Stop command after a button press
on a Mycroft unit
        #. ``EnclosureReader`` captures the Stop command
        #. Notify all Mycroft Core processes (e.g. skills) to be
stopped
```

```
    Note: A command is identified by a line break
    """

    def __init__(self, serial, ws):
        super(EnclosureReader, self).__init__(target=self.read)
        self.alive = True
        self.daemon = True
        self.serial = serial
        self.ws = ws
        self.start()

    def read(self):
        while self.alive:
            try:
                data = self.serial.readline()[:-2]
                if data:
                    self.process(data)
                    LOG.info("Reading: " + data)
            except Exception as e:
                LOG.error("Reading error: {0}".format(e))

    def process(self, data):
        self.ws.emit(Message(data))

        if "Command: system.version" in data:
            self.ws.emit(Message("enclosure.start"))

        if "mycroft.stop" in data:
            create_signal('buttonPress')  # FIXME - Must use WS
instead
            self.ws.emit(Message("mycroft.stop"))

        if "volume.up" in data:
            self.ws.emit(
                Message("IncreaseVolumeIntent", {'play_sound':
True}))

        if "volume.down" in data:
            self.ws.emit(
                Message("DecreaseVolumeIntent", {'play_sound':
True}))

        if "system.test.begin" in data:
            self.ws.emit(Message('recognizer_loop:sleep'))

        if "system.test.end" in data:
            self.ws.emit(Message('recognizer_loop:wake_up'))

        if "mic.test" in data:
            mixer = Mixer()
            prev_vol = mixer.getvolume()[0]
            mixer.setvolume(35)
            self.ws.emit(Message("speak", {
                'utterance': "I am testing one two three"}))

            time.sleep(0.5)  # Prevents recording the loud button
press
            record("/tmp/test.wav", 3.0)
            mixer.setvolume(prev_vol)
            play_wav("/tmp/test.wav").communicate()

            # Test audio muting on arduino
            subprocess.call('speaker-test -P 10 -l 0 -s 1',
shell=True)

        if "unit.shutdown" in data:
            self.ws.emit(
                Message("enclosure.eyes.timedspin",
                        {'length': 12000}))
            self.ws.emit(Message("enclosure.mouth.reset"))
            subprocess.call('systemctl poweroff -i', shell=True)

        if "unit.reboot" in data:
            self.ws.emit(
                Message("enclosure.eyes.spin"))
            self.ws.emit(Message("enclosure.mouth.reset"))
            subprocess.call('systemctl reboot -i', shell=True)

        if "unit.setwifi" in data:
            self.ws.emit(Message("mycroft.wifi.start"))

        if "unit.factory-reset" in data:
            subprocess.call(
                'rm ~/.mycroft/identity/identity.json',
                shell=True)
            self.ws.emit(
                Message("enclosure.eyes.spin"))
            self.ws.emit(Message("enclosure.mouth.reset"))
            subprocess.call('systemctl reboot -i', shell=True)

    def stop(self):
        self.alive = False


class EnclosureWriter(Thread):
    """
    Writes data to Serial port.
        #. Enqueues all commands received from Mycroft
enclosures
           implementation
        #. Process them on the received order by writing on the
Serial port

    E.g. Displaying a text on Mycroft's Mouth
        #. ``EnclosureMouth`` sends a text command
        #. ``EnclosureWriter`` captures and enqueue the
command
        #. ``EnclosureWriter`` removes the next command from
the queue
        #. ``EnclosureWriter`` writes the command to Serial port

    Note: A command has to end with a line break
    """

    def __init__(self, serial, ws, size=16):
        super(EnclosureWriter, self).__init__(target=self.flush)
        self.alive = True
        self.daemon = True
        self.serial = serial
        self.ws = ws
        self.commands = Queue(size)
        self.start()

    def flush(self):
        while self.alive:
            try:
                cmd = self.commands.get()
                self.serial.write(cmd + '\n')
                LOG.info("Writing: " + cmd)
                self.commands.task_done()
            except Exception as e:
```

```
            LOG.error("Writing error: {0}".format(e))

    def write(self, command):
        self.commands.put(str(command))

    def stop(self):
        self.alive = False


class Enclosure(object):
    """
    Serves as a communication interface between Arduino
and Mycroft Core.

    ``Enclosure`` initializes and aggregates all enclosures
implementation.

    E.g. ``EnclosureEyes``, ``EnclosureMouth`` and
``EnclosureArduino``

    It also listens to the basis events in order to perform those
core actions
    on the unit.

    E.g. Start and Stop talk animation
    """

    def __init__(self):
        self.ws = WebsocketClient()
        ConfigurationManager.init(self.ws)
        self.config =
ConfigurationManager.get().get("enclosure")
        self.__init_serial()
        self.reader = EnclosureReader(self.serial, self.ws)
        self.writer = EnclosureWriter(self.serial, self.ws)
        self.writer.write("system.version")
        self.ws.on("enclosure.start", self.start)
        self.started = False
        Timer(5, self.stop).start()

    def start(self, event=None):
        self.eyes = EnclosureEyes(self.ws, self.writer)
        self.mouth = EnclosureMouth(self.ws, self.writer)
        self.system = EnclosureArduino(self.ws, self.writer)
        self.weather = EnclosureWeather(self.ws, self.writer)
        self.__register_events()
        self.started = True

    def __init_serial(self):
        try:
            self.port = self.config.get("port")
            self.rate = self.config.get("rate")
            self.timeout = self.config.get("timeout")
            self.serial = serial.serial_for_url(
                url=self.port, baudrate=self.rate,
timeout=self.timeout)
            LOG.info("Connected to: %s rate: %s timeout: %s" %
                     (self.port, self.rate, self.timeout))
        except:
            LOG.error("Impossible to connect to serial port: " +
self.port)
            raise

    def __register_events(self):
        self.ws.on('enclosure.mouth.events.activate',
                   self.__register_mouth_events)
        self.ws.on('enclosure.mouth.events.deactivate',
                   self.__remove_mouth_events)
        self.__register_mouth_events()

    def __register_mouth_events(self, event=None):
        self.ws.on('recognizer_loop:record_begin',
self.mouth.listen)
        self.ws.on('recognizer_loop:record_end',
self.mouth.reset)
        self.ws.on('recognizer_loop:audio_output_start',
self.mouth.talk)
        self.ws.on('recognizer_loop:audio_output_end',
self.mouth.reset)

    def __remove_mouth_events(self, event=None):
        self.ws.remove('recognizer_loop:record_begin',
self.mouth.listen)
        self.ws.remove('recognizer_loop:record_end',
self.mouth.reset)
        self.ws.remove('recognizer_loop:audio_output_start',
                       self.mouth.talk)
        self.ws.remove('recognizer_loop:audio_output_end',
                       self.mouth.reset)

    def speak(self, text):
        self.ws.emit(Message("speak", {'utterance': text}))

    def run(self):
        try:
            self.ws.run_forever()
        except Exception as e:
            LOG.error("Error: {0}".format(e))
            self.stop()

    def stop(self):
        if not self.started:
            self.writer.stop()
            self.reader.stop()
            self.serial.close()
            self.ws.close()
```

Directory: intelora-core/mycroft/client/enclore/api.py

```
from mycroft.messagebus.message import Message
from mycroft.util.log import getLogger

__author__ = 'jdorleans'

LOGGER = getLogger(__name__)


class EnclosureAPI:
    """
    This API is intended to be used to interface with the
hardware
    that is running Mycroft.  It exposes all possible commands
which
    can be sent to a Mycroft enclosure implementation.
    """

    def __init__(self, ws):
        self.ws = ws

    def system_reset(self):
```

```
        self.ws.emit(Message("enclosure.system.reset"))

    def system_mute(self):
        self.ws.emit(Message("enclosure.system.mute"))

    def system_unmute(self):
        self.ws.emit(Message("enclosure.system.unmute"))

    def system_blink(self, times):
        self.ws.emit(Message("enclosure.system.blink", {'times':
times}))

    def eyes_on(self):
        self.ws.emit(Message("enclosure.eyes.on"))

    def eyes_off(self):
        self.ws.emit(Message("enclosure.eyes.off"))

    def eyes_blink(self, side):
        self.ws.emit(Message("enclosure.eyes.blink", {'side':
side}))

    def eyes_narrow(self):
        self.ws.emit(Message("enclosure.eyes.narrow"))

    def eyes_look(self, side):
        self.ws.emit(Message("enclosure.eyes.look", {'side':
side}))

    def eyes_color(self, r=255, g=255, b=255):
        self.ws.emit(Message("enclosure.eyes.color",
                             {'r': r, 'g': g, 'b': b}))

    def eyes_brightness(self, level=30):
        self.ws.emit(Message("enclosure.eyes.level", {'level':
level}))

    def eyes_reset(self):
        self.ws.emit(Message("enclosure.eyes.reset"))

    def eyes_timed_spin(self, length):
        self.ws.emit(Message("enclosure.eyes.timedspin",
                             {'length': length}))

    def eyes_volume(self, volume):
        self.ws.emit(Message("enclosure.eyes.volume",
{'volume': volume}))

    def mouth_reset(self):
        self.ws.emit(Message("enclosure.mouth.reset"))

    def mouth_talk(self):
        self.ws.emit(Message("enclosure.mouth.talk"))

    def mouth_think(self):
        self.ws.emit(Message("enclosure.mouth.think"))

    def mouth_listen(self):
        self.ws.emit(Message("enclosure.mouth.listen"))

    def mouth_smile(self):
        self.ws.emit(Message("enclosure.mouth.smile"))

    def mouth_viseme(self, code):
        self.ws.emit(Message("enclosure.mouth.viseme",
{'code': code}))

    def mouth_text(self, text=""):
        self.ws.emit(Message("enclosure.mouth.text", {'text':
text}))

    def weather_display(self, img_code, temp):
        self.ws.emit(Message("enclosure.weather.display",
                             {'img_code': img_code, 'temp': temp}))

    def activate_mouth_events(self):

self.ws.emit(Message('enclosure.mouth.events.activate'))

    def deactivate_mouth_events(self):

self.ws.emit(Message('enclosure.mouth.events.deactivate'))
```

Directory: intelora-core/mycroft/client/enclore/arduino.py

```
from mycroft.util.log import getLogger

__author__ = 'jdorleans'

LOGGER = getLogger(__name__)


class EnclosureArduino:
    """
    Listens to enclosure commands for Mycroft's Arduino.

    Performs the associated command on Arduino by writing
on the Serial port.
    """

    def __init__(self, ws, writer):
        self.ws = ws
        self.writer = writer
        self.__init_events()

    def __init_events(self):
        self.ws.on('enclosure.system.reset', self.reset)
        self.ws.on('enclosure.system.mute', self.mute)
        self.ws.on('enclosure.system.unmute', self.unmute)
        self.ws.on('enclosure.system.blink', self.blink)

    def reset(self, event=None):
        self.writer.write("system.reset")

    def mute(self, event=None):
        self.writer.write("system.mute")

    def unmute(self, event=None):
        self.writer.write("system.unmute")

    def blink(self, event=None):
        times = 1
        if event and event.data:
            times = event.data.get("times", times)
        self.writer.write("system.blink=" + str(times))
```

Directory: intelora-core/mycroft/client/enclore/eyes.py

```
from mycroft.util.log import getLogger

__author__ = 'jdorleans'

LOGGER = getLogger(__name__)


class EnclosureEyes:
    """
    Listens to enclosure commands for Mycroft's Eyes.

    Performs the associated command on Arduino by writing
on the Serial port.
    """

    def __init__(self, ws, writer):
        self.ws = ws
        self.writer = writer
        self.__init_events()

    def __init_events(self):
        self.ws.on('enclosure.eyes.on', self.on)
        self.ws.on('enclosure.eyes.off', self.off)
        self.ws.on('enclosure.eyes.blink', self.blink)
        self.ws.on('enclosure.eyes.narrow', self.narrow)
        self.ws.on('enclosure.eyes.look', self.look)
        self.ws.on('enclosure.eyes.color', self.color)
        self.ws.on('enclosure.eyes.level', self.brightness)
        self.ws.on('enclosure.eyes.volume', self.volume)
        self.ws.on('enclosure.eyes.spin', self.spin)
        self.ws.on('enclosure.eyes.timedspin', self.timed_spin)
        self.ws.on('enclosure.eyes.reset', self.reset)

    def on(self, event=None):
        self.writer.write("eyes.on")

    def off(self, event=None):
        self.writer.write("eyes.off")

    def blink(self, event=None):
        side = "b"
        if event and event.data:
            side = event.data.get("side", side)
        self.writer.write("eyes.blink=" + side)

    def narrow(self, event=None):
        self.writer.write("eyes.narrow")

    def look(self, event=None):
        if event and event.data:
            side = event.data.get("side", "")
            self.writer.write("eyes.look=" + side)

    def color(self, event=None):
        r, g, b = 255, 255, 255
        if event and event.data:
            r = int(event.data.get("r"), r)
            g = int(event.data.get("g"), g)
            b = int(event.data.get("b"), b)
        color = (r * 65536) + (g * 256) + b
        self.writer.write("eyes.color=" + str(color))

    def brightness(self, event=None):
        level = 30
        if event and event.data:
            level = event.data.get("level", level)
        self.writer.write("eyes.level=" + str(level))

    def volume(self, event=None):
        volume = 4
        if event and event.data:
            volume = event.data.get("volume", volume)
        self.writer.write("eyes.volume=" + str(volume))

    def reset(self, event=None):
        self.writer.write("eyes.reset")

    def spin(self, event=None):
        self.writer.write("eyes.spin")

    def timed_spin(self, event=None):
        length = 5000
        if event and event.data:
            length = event.data.get("length", length)
        self.writer.write("eyes.spin=" + str(length))
```

Directory: intelora-core/mycroft/client/enclore/main.py

```
import sys

from mycroft.client.enclosure import Enclosure


def main():
    enclosure = Enclosure()
    try:
        enclosure.run()
    except Exception as e:
        print(e)
    finally:
        sys.exit()


if __name__ == "__main__":
    main()
```

Directory: intelora-core/mycroft/client/enclore/mouth.py

```
from mycroft.util.log import getLogger

__author__ = 'jdorleans'

LOG = getLogger(__name__)


class EnclosureMouth:
    """
    Listens to enclosure commands for Mycroft's Mouth.

    Performs the associated command on Arduino by writing
on the Serial port.
    """

    def __init__(self, ws, writer):
        self.ws = ws
        self.writer = writer
        self.__init_events()

    def __init_events(self):
```

```
        self.ws.on('enclosure.mouth.reset', self.reset)
        self.ws.on('enclosure.mouth.talk', self.talk)
        self.ws.on('enclosure.mouth.think', self.think)
        self.ws.on('enclosure.mouth.listen', self.listen)
        self.ws.on('enclosure.mouth.smile', self.smile)
        self.ws.on('enclosure.mouth.viseme', self.viseme)
        self.ws.on('enclosure.mouth.text', self.text)

    def reset(self, event=None):
        self.writer.write("mouth.reset")

    def talk(self, event=None):
        self.writer.write("mouth.talk")

    def think(self, event=None):
        self.writer.write("mouth.think")

    def listen(self, event=None):
        self.writer.write("mouth.listen")

    def smile(self, event=None):
        self.writer.write("mouth.smile")

    def viseme(self, event=None):
        if event and event.data:
            code = event.data.get("code")
            if code:
                self.writer.write("mouth.viseme=" + code)

    def text(self, event=None):
        text = ""
        if event and event.data:
            text = event.data.get("text", text)
        self.writer.write("mouth.text=" + text)
```

Directory: intelora-core/mycroft/client/enclore/weather.py

```
from mycroft.util.log import getLogger

__author__ = 'iward'

LOGGER = getLogger(__name__)


class EnclosureWeather:
    """
    Listens to enclosure commands to display indicators of the
weather.

    Performs the associated command on Arduino by writing
on the Serial port.
    """

    def __init__(self, ws, writer):
        self.ws = ws
        self.writer = writer
        self.__init_events()

    def __init_events(self):
        self.ws.on('enclosure.weather.display', self.display)

    def display(self, event=None):
        if event and event.data:
            img_code = event.data.get("img_code", None)
            temp = event.data.get("temp", None)
            if img_code is not None and temp is not None:
                msg = "weather.display=" + str(img_code) +
str(temp)
                self.writer.write(msg)
```

Directory: intelora-core/mycroft/client/speech/listener.py

```
import time
from Queue import Queue
from threading import Thread

import speech_recognition as sr
from pyee import EventEmitter
from requests import HTTPError

from mycroft.client.speech.local_recognizer import
LocalRecognizer
from mycroft.client.speech.mic import MutableMicrophone,
ResponsiveRecognizer
from mycroft.configuration import ConfigurationManager
from mycroft.messagebus.message import Message
from mycroft.metrics import MetricsAggregator
from mycroft.session import SessionManager
from mycroft.stt import STTFactory
from mycroft.util import connected
from mycroft.util.log import getLogger

LOG = getLogger(__name__)


class AudioProducer(Thread):
    """
    AudioProducer
    given a mic and a recognizer implementation,
continuously listens to the
    mic for potential speech chunks and pushes them onto
the queue.
    """

    def __init__(self, state, queue, mic, recognizer, emitter):
        super(AudioProducer, self).__init__()
        self.daemon = True
        self.state = state
        self.queue = queue
        self.mic = mic
        self.recognizer = recognizer
        self.emitter = emitter

    def run(self):
        with self.mic as source:
            self.recognizer.adjust_for_ambient_noise(source)
            while self.state.running:
                try:
                    audio = self.recognizer.listen(source, self.emitter)
                    self.queue.put(audio)
                except IOError, ex:
                    # NOTE: Audio stack on raspi is slightly different,
throws
                    # IOError every other listen, almost like it can't
handle
                    # buffering audio between listen loops.
                    # The internet was not helpful.
```

```
                #
http://stackoverflow.com/questions/10733903/pyaudio-
input-overflowed
                self.emitter.emit("recognizer_loop:ioerror", ex)


class AudioConsumer(Thread):
    """
    AudioConsumer
    Consumes AudioData chunks off the queue
    """

    # In seconds, the minimum audio size to be sent to
remote STT
    MIN_AUDIO_SIZE = 0.5

    def __init__(self, state, queue, emitter, stt,
                 wakeup_recognizer, mycroft_recognizer):
        super(AudioConsumer, self).__init__()
        self.daemon = True
        self.queue = queue
        self.state = state
        self.emitter = emitter
        self.stt = stt
        self.wakeup_recognizer = wakeup_recognizer
        self.mycroft_recognizer = mycroft_recognizer
        self.metrics = MetricsAggregator()

    def run(self):
        while self.state.running:
            self.read()

    def read(self):
        audio = self.queue.get()

        if self.state.sleeping:
            self.wake_up(audio)
        else:
            self.process(audio)

    # TODO: Localization
    def wake_up(self, audio):
        if
self.wakeup_recognizer.is_recognized(audio.frame_data,
                                     self.metrics):
            SessionManager.touch()
            self.state.sleeping = False
            self.__speak("I'm awake.")
            self.metrics.increment("mycroft.wakeup")

    @staticmethod
    def _audio_length(audio):
        return float(len(audio.frame_data)) / (
            audio.sample_rate * audio.sample_width)

    # TODO: Localization
    def process(self, audio):
        SessionManager.touch()
        payload = {
            'utterance': self.mycroft_recognizer.key_phrase,
            'session': SessionManager.get().session_id,
        }
        self.emitter.emit("recognizer_loop:wakeword",
payload)

        if self._audio_length(audio) < self.MIN_AUDIO_SIZE:
            LOG.warn("Audio too short to be processed")
        elif connected():
            self.transcribe(audio)
        else:
            self.__speak("Mycroft seems not to be connected to
the Internet")

    def transcribe(self, audio):
        text = None
        try:
            text = self.stt.execute(audio).lower().strip()
            LOG.debug("STT: " + text)
        except sr.RequestError as e:
            LOG.error("Could not request Speech Recognition
{0}".format(e))
        except HTTPError as e:
            if e.response.status_code == 401:
                text = "pair my device"
                LOG.warn("Access Denied at mycroft.ai")
        except Exception as e:
            LOG.error(e)
            LOG.error("Speech Recognition could not understand
audio")
            self.__speak("Sorry, I didn't catch that")
        if text:
            payload = {
                'utterances': [text],
                'session': SessionManager.get().session_id
            }
            self.emitter.emit("recognizer_loop:utterance",
payload)
            self.metrics.attr('utterances', [text])

    def __speak(self, utterance):
        payload = {
            'utterance': utterance,
            'session': SessionManager.get().session_id
        }
        self.emitter.emit("speak", Message("speak", payload))


class RecognizerLoopState(object):
    def __init__(self):
        self.running = False
        self.sleeping = False


class RecognizerLoop(EventEmitter):
    def __init__(self):
        super(RecognizerLoop, self).__init__()
        config = ConfigurationManager.get()
        lang = config.get('lang')
        self.config = config.get('listener')
        rate = self.config.get('sample_rate')
        device_index = self.config.get('device_index')

        self.microphone = MutableMicrophone(device_index,
rate)
        # FIXME - channels are not been used
        self.microphone.CHANNELS = self.config.get('channels')
        self.mycroft_recognizer =
self.create_mycroft_recognizer(rate, lang)
        # TODO - localization
        self.wakeup_recognizer =
self.create_wakeup_recognizer(rate, lang)
```

```
        self.remote_recognizer =
ResponsiveRecognizer(self.mycroft_recognizer)
        self.state = RecognizerLoopState()

    def create_mycroft_recognizer(self, rate, lang):
        wake_word = self.config.get('wake_word')
        phonemes = self.config.get('phonemes')
        threshold = self.config.get('threshold')
        return LocalRecognizer(wake_word, phonemes,
threshold, rate, lang)

    @staticmethod
    def create_wakeup_recognizer(rate, lang):
        return LocalRecognizer("wake up", "W EY K . AH P", 1e-
10, rate, lang)

    def start_async(self):
        self.state.running = True
        queue = Queue()
        AudioProducer(self.state, queue, self.microphone,
                      self.remote_recognizer, self).start()
        AudioConsumer(self.state, queue, self,
STTFactory.create(),
                      self.wakeup_recognizer,
self.mycroft_recognizer).start()

    def stop(self):
        self.state.running = False

    def mute(self):
        if self.microphone:
            self.microphone.mute()

    def unmute(self):
        if self.microphone:
            self.microphone.unmute()

    def sleep(self):
        self.state.sleeping = True

    def awaken(self):
        self.state.sleeping = False

    def run(self):
        self.start_async()
        while self.state.running:
            try:
                time.sleep(1)
            except KeyboardInterrupt as e:
                LOG.error(e)
                self.stop()
```

Directory: intelora-core/mycroft/client/speech/local_recognizer.py

```
import os
import tempfile
import time
from os.path import join, dirname, abspath

from pocketsphinx import Decoder

__author__ = 'seanfitz, jdorleans'

BASEDIR = dirname(abspath(__file__))


class LocalRecognizer(object):
    def __init__(self, key_phrase, phonemes, threshold,
sample_rate=16000,
                 lang="en-us"):
        self.lang = str(lang)
        self.key_phrase = str(key_phrase)
        self.sample_rate = sample_rate
        self.threshold = threshold
        self.phonemes = phonemes
        dict_name = self.create_dict(key_phrase, phonemes)
        self.decoder = Decoder(self.create_config(dict_name))

    def create_dict(self, key_phrase, phonemes):
        (fd, file_name) = tempfile.mkstemp()
        words = key_phrase.split()
        phoneme_groups = phonemes.split('.')
        with os.fdopen(fd, 'w') as f:
            for word, phoneme in zip(words, phoneme_groups):
                f.write(word + ' ' + phoneme + '\n')
        return file_name

    def create_config(self, dict_name):
        config = Decoder.default_config()
        config.set_string('-hmm', join(BASEDIR, 'model',
self.lang, 'hmm'))
        config.set_string('-dict', dict_name)
        config.set_string('-keyphrase', self.key_phrase)
        config.set_float('-kws_threshold', self.threshold)
        config.set_float('-samprate', self.sample_rate)
        config.set_int('-nfft', 2048)
        config.set_string('-logfn', '/dev/null')
        return config

    def transcribe(self, byte_data, metrics=None):
        start = time.time()
        self.decoder.start_utt()
        self.decoder.process_raw(byte_data, False, False)
        self.decoder.end_utt()
        if metrics:
            metrics.timer("mycroft.stt.local.time_s", time.time() -
start)
        return self.decoder.hyp()

    def is_recognized(self, byte_data, metrics):
        hyp = self.transcribe(byte_data, metrics)
        return hyp and self.key_phrase in hyp.hypstr.lower()

    def found_wake_word(self, hypothesis):
        return hypothesis and self.key_phrase in
hypothesis.hypstr.lower()
```

Directory: intelora-core/mycroft/client/speech/main.py

```
import re
import sys
from threading import Thread, Lock

from mycroft.client.enclosure.api import EnclosureAPI
from mycroft.client.speech.listener import RecognizerLoop
from mycroft.configuration import ConfigurationManager
from mycroft.identity import IdentityManager
```

```
from mycroft.messagebus.client.ws import WebsocketClient
from mycroft.messagebus.message import Message
from mycroft.tts import TTSFactory
from mycroft.util import kill
from mycroft.util.log import getLogger

logger = getLogger("SpeechClient")
ws = None
tts = TTSFactory.create()
lock = Lock()
loop = None

config = ConfigurationManager.get()


def handle_record_begin():
    logger.info("Begin Recording...")
    ws.emit(Message('recognizer_loop:record_begin'))


def handle_record_end():
    logger.info("End Recording...")
    ws.emit(Message('recognizer_loop:record_end'))


def handle_wakeword(event):
    logger.info("Wakeword Detected: " + event['utterance'])
    ws.emit(Message('recognizer_loop:wakeword', event))


def handle_utterance(event):
    logger.info("Utterance: " + str(event['utterances']))
    ws.emit(Message('recognizer_loop:utterance', event))


def mute_and_speak(utterance):
    lock.acquire()
    ws.emit(Message("recognizer_loop:audio_output_start"))
    try:
        logger.info("Speak: " + utterance)
        loop.mute()
        tts.execute(utterance)
    finally:
        loop.unmute()
        lock.release()

ws.emit(Message("recognizer_loop:audio_output_end"))


def handle_multi_utterance_intent_failure(event):
    logger.info("Failed to find intent on multiple intents.")
    # TODO: Localize
    mute_and_speak("Sorry, I didn't catch that. Please
rephrase your request.")


def handle_speak(event):
    utterance = event.data['utterance']
    chunks = re.split(r'(?<!\w\.\w.)(?<![A-Z][a-
z]\.)(?<=\.|\?)\s', utterance)
    for chunk in chunks:
        mute_and_speak(chunk)


def handle_sleep(event):
    loop.sleep()


def handle_wake_up(event):
    loop.awaken()


def handle_stop(event):
    kill([config.get('tts').get('module')])
    kill(["aplay"])


def handle_paired(event):
    IdentityManager.update(event.data)


def handle_open():
    EnclosureAPI(ws).system_reset()


def connect():
    ws.run_forever()


def main():
    global ws
    global loop
    ws = WebsocketClient()
    tts.init(ws)
    ConfigurationManager.init(ws)
    loop = RecognizerLoop()
    loop.on('recognizer_loop:utterance', handle_utterance)
    loop.on('recognizer_loop:record_begin',
handle_record_begin)
    loop.on('recognizer_loop:wakeword', handle_wakeword)
    loop.on('recognizer_loop:record_end',
handle_record_end)
    loop.on('speak', handle_speak)
    ws.on('open', handle_open)
    ws.on('speak', handle_speak)
    ws.on(
        'multi_utterance_intent_failure',
        handle_multi_utterance_intent_failure)
    ws.on('recognizer_loop:sleep', handle_sleep)
    ws.on('recognizer_loop:wake_up', handle_wake_up)
    ws.on('mycroft.stop', handle_stop)
    ws.on("mycroft.paired", handle_paired)
    event_thread = Thread(target=connect)
    event_thread.setDaemon(True)
    event_thread.start()

    try:
        loop.run()
    except KeyboardInterrupt, e:
        logger.exception(e)
        event_thread.exit()
        sys.exit()


if __name__ == "__main__":
    main()
```

Directory: intelora-core/mycroft/client/speech/mic.py

```
import audioop
```

```
import collections
from time import sleep

import pyaudio
import speech_recognition
from speech_recognition import (
    Microphone,
    AudioSource,
    AudioData
)

from mycroft.configuration import ConfigurationManager
from mycroft.util import check_for_signal
from mycroft.util.log import getLogger
from mycroft.util import play_wav

listener_config = ConfigurationManager.get().get('listener')
logger = getLogger(__name__)
__author__ = 'seanfitz'


class MutableStream(object):
    def __init__(self, wrapped_stream, format, muted=False):
        assert wrapped_stream is not None
        self.wrapped_stream = wrapped_stream
        self.muted = muted
        self.SAMPLE_WIDTH =
pyaudio.get_sample_size(format)
        self.muted_buffer = b''.join([b'\x00' *
self.SAMPLE_WIDTH])

    def mute(self):
        self.muted = True

    def unmute(self):
        self.muted = False

    def read(self, size):
        frames = collections.deque()
        remaining = size
        while remaining > 0:
            to_read =
min(self.wrapped_stream.get_read_available(), remaining)
            if to_read == 0:
                sleep(.01)
                continue
            result = self.wrapped_stream.read(to_read)
            frames.append(result)
            remaining -= to_read

        if self.muted:
            return self.muted_buffer
        input_latency =
self.wrapped_stream.get_input_latency()
        if input_latency > 0.2:
            logger.warn("High input latency: %f" % input_latency)
        audio = b"".join(list(frames))
        return audio

    def close(self):
        self.wrapped_stream.close()
        self.wrapped_stream = None

    def is_stopped(self):
        return self.wrapped_stream.is_stopped()

    def stop_stream(self):
        return self.wrapped_stream.stop_stream()


class MutableMicrophone(Microphone):
    def __init__(self, device_index=None,
sample_rate=16000, chunk_size=1024):
        Microphone.__init__(
            self, device_index=device_index,
sample_rate=sample_rate,
            chunk_size=chunk_size)
        self.muted = False

    def __enter__(self):
        assert self.stream is None, \
            "This audio source is already inside a context
manager"
        self.audio = pyaudio.PyAudio()
        self.stream = MutableStream(self.audio.open(
            input_device_index=self.device_index, channels=1,
            format=self.format, rate=self.SAMPLE_RATE,
            frames_per_buffer=self.CHUNK,
            input=True,  # stream is an input stream
        ), self.format, self.muted)
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        if not self.stream.is_stopped():
            self.stream.stop_stream()
        self.stream.close()
        self.stream = None
        self.audio.terminate()

    def mute(self):
        self.muted = True
        if self.stream:
            self.stream.mute()

    def unmute(self):
        self.muted = False
        if self.stream:
            self.stream.unmute()


class
ResponsiveRecognizer(speech_recognition.Recognizer):
    # The maximum audio in seconds to keep for transcribing
a phrase
    # The wake word must fit in this time
    SAVED_WW_SEC = 1.0

    # Padding of silence when feeding to pocketsphinx
    SILENCE_SEC = 0.01

    # The minimum seconds of noise before a
    # phrase can be considered complete
    MIN_LOUD_SEC_PER_PHRASE = 0.1

    # The maximum length a phrase can be recorded,
    # provided there is noise the entire time
    RECORDING_TIMEOUT = 30.0

    # The maximum time it will continue to record silence
    # when not enough noise has been detected
    RECORDING_TIMEOUT_WITH_SILENCE = 3.0
```

```
    # Time between pocketsphinx checks for the wake word
    SEC_BETWEEN_WW_CHECKS = 0.2

    def __init__(self, wake_word_recognizer):
        speech_recognition.Recognizer.__init__(self)
        self.wake_word_recognizer = wake_word_recognizer
        self.audio = pyaudio.PyAudio()
        self.multiplier = listener_config.get('multiplier')
        self.energy_ratio = listener_config.get('energy_ratio')

    @staticmethod
    def record_sound_chunk(source):
        return source.stream.read(source.CHUNK)

    @staticmethod
    def calc_energy(sound_chunk, sample_width):
        return audioop.rms(sound_chunk, sample_width)

    def wake_word_in_audio(self, frame_data):
        hyp =
self.wake_word_recognizer.transcribe(frame_data)
        return
self.wake_word_recognizer.found_wake_word(hyp)

    def record_phrase(self, source, sec_per_buffer):
        """
        This attempts to record an entire spoken phrase.
Essentially,
        this waits for a period of silence and then returns the
audio

        :rtype: bytearray
        :param source: AudioSource
        :param sec_per_buffer: Based on source.SAMPLE_RATE
        :return: bytearray representing the frame_data of the
recorded phrase
        """
        num_loud_chunks = 0
        noise = 0

        max_noise = 25
        min_noise = 0

        def increase_noise(level):
            if level < max_noise:
                return level + 200 * sec_per_buffer
            return level

        def decrease_noise(level):
            if level > min_noise:
                return level - 100 * sec_per_buffer
            return level

        # Smallest number of loud chunks required to return
        min_loud_chunks =
int(self.MIN_LOUD_SEC_PER_PHRASE / sec_per_buffer)

        # Maximum number of chunks to record before timing
out
        max_chunks = int(self.RECORDING_TIMEOUT /
sec_per_buffer)
        num_chunks = 0

        # Will return if exceeded this even if there's not enough
loud chunks
        max_chunks_of_silence =
int(self.RECORDING_TIMEOUT_WITH_SILENCE /
                                 sec_per_buffer)

        # bytearray to store audio in
        byte_data = '\0' * source.SAMPLE_WIDTH

        phrase_complete = False
        while num_chunks < max_chunks and not
phrase_complete:
            chunk = self.record_sound_chunk(source)
            byte_data += chunk
            num_chunks += 1

            energy = self.calc_energy(chunk,
source.SAMPLE_WIDTH)
            test_threshold = self.energy_threshold *
self.multiplier
            is_loud = energy > test_threshold
            if is_loud:
                noise = increase_noise(noise)
                num_loud_chunks += 1
            else:
                noise = decrease_noise(noise)
                self.adjust_threshold(energy, sec_per_buffer)

            was_loud_enough = num_loud_chunks >
min_loud_chunks
            quiet_enough = noise <= min_noise
            recorded_too_much_silence = num_chunks >
max_chunks_of_silence
            if quiet_enough and (was_loud_enough or
recorded_too_much_silence):
                phrase_complete = True
            if check_for_signal('buttonPress'):
                phrase_complete = True

        return byte_data

    @staticmethod
    def sec_to_bytes(sec, source):
        return sec * source.SAMPLE_RATE *
source.SAMPLE_WIDTH

    def wait_until_wake_word(self, source, sec_per_buffer):
        num_silent_bytes = int(self.SILENCE_SEC *
source.SAMPLE_RATE *
                               source.SAMPLE_WIDTH)

        silence = '\0' * num_silent_bytes

        # bytearray to store audio in
        byte_data = silence

        buffers_per_check = self.SEC_BETWEEN_WW_CHECKS /
sec_per_buffer
        buffers_since_check = 0.0

        # Max bytes for byte_data before audio is removed
from the front
        max_size = self.sec_to_bytes(self.SAVED_WW_SEC,
source)

        said_wake_word = False
        while not said_wake_word:
            if check_for_signal('buttonPress'):
```

```
            said_wake_word = True
            continue

        chunk = self.record_sound_chunk(source)

        energy = self.calc_energy(chunk,
source.SAMPLE_WIDTH)
        if energy < self.energy_threshold * self.multiplier:
            self.adjust_threshold(energy, sec_per_buffer)

        needs_to_grow = len(byte_data) < max_size
        if needs_to_grow:
            byte_data += chunk
        else:  # Remove beginning of audio and add new
chunk to end
            byte_data = byte_data[len(chunk):] + chunk

        buffers_since_check += 1.0
        if buffers_since_check < buffers_per_check:
            buffers_since_check -= buffers_per_check
            said_wake_word =
self.wake_word_in_audio(byte_data + silence)

    @staticmethod
    def create_audio_data(raw_data, source):
        """
        Constructs an AudioData instance with the same
parameters
        as the source and the specified frame_data
        """
        return AudioData(raw_data, source.SAMPLE_RATE,
source.SAMPLE_WIDTH)

    def listen(self, source, emitter):
        """
        Listens for audio that Mycroft should respond to

        :param source: an ``AudioSource`` instance for reading
from
        :param emitter: a pyee EventEmitter for sending when
the wakeword
                        has been found
        """
        assert isinstance(source, AudioSource), "Source must be
an AudioSource"

        bytes_per_sec = source.SAMPLE_RATE *
source.SAMPLE_WIDTH
        sec_per_buffer = float(source.CHUNK) / bytes_per_sec

        logger.debug("Waiting for wake word...")
        self.wait_until_wake_word(source, sec_per_buffer)

        play_wav('/tmp/beep_hi.wav')
        logger.debug("Listening...")

        emitter.emit("recognizer_loop:record_begin")
        frame_data = self.record_phrase(source,
sec_per_buffer)
        audio_data = self.create_audio_data(frame_data,
source)
        emitter.emit("recognizer_loop:record_end")

        play_wav('/tmp/beep_lo.wav')
        logger.debug("Thinking...")

        return audio_data

    def adjust_threshold(self, energy, seconds_per_buffer):
        if self.dynamic_energy_threshold and energy > 0:
            # account for different chunk sizes and rates
            damping = (
                self.dynamic_energy_adjustment_damping **
seconds_per_buffer)
            target_energy = energy * self.energy_ratio
            self.energy_threshold = (
                self.energy_threshold * damping +
                target_energy * (1 - damping))
```

Directory: intelora-core/mycroft/client/speech/recognizer_wrapper.py

```
import json

import requests
from speech_recognition import UnknownValueError

from mycroft.configuration import ConfigurationManager
from mycroft.identity import IdentityManager
from mycroft.metrics import Stopwatch
from mycroft.util import CerberusAccessDenied
from mycroft.util.log import getLogger
from mycroft.util.setup_base import get_version

__author__ = 'seanfitz'

log = getLogger("RecognizerWrapper")

config = ConfigurationManager.get().get('speech_client')


class GoogleRecognizerWrapper(object):
    def __init__(self, recognizer):
        self.recognizer = recognizer

    def transcribe(
            self, audio, language="en-US", show_all=False,
metrics=None):
        key = config.get('goog_api_key')
        return self.recognizer.recognize_google(
            audio, key=key, language=language,
show_all=show_all)


class WitRecognizerWrapper(object):
    def __init__(self, recognizer):
        self.recognizer = recognizer

    def transcribe(
            self, audio, language="en-US", show_all=False,
metrics=None):
        assert language == "en-US", \
            "language must be default, language parameter not
supported."
        key = config.get('wit_api_key')
        return self.recognizer.recognize_wit(audio, key,
show_all=show_all)


class IBMRecognizerWrapper(object):
```

```
  def __init__(self, recognizer):
    self.recognizer = recognizer

  def transcribe(
      self, audio, language="en-US", show_all=False,
metrics=None):
    username = config.get('ibm_username')
    password = config.get('ibm_password')
    return self.recognizer.recognize_ibm(
      audio, username, password, language=language,
show_all=show_all)


class CerberusGoogleProxy(object):
  def __init__(self, _):
    self.version = get_version()

  def transcribe(
      self, audio, language="en-US", show_all=False,
metrics=None):
    timer = Stopwatch()
    timer.start()
    identity = IdentityManager().get()
    headers = {}
    if identity.token:
      headers['Authorization'] = 'Bearer %s:%s' % (
        identity.device_id, identity.token)

    response = requests.post(config.get("proxy_host") +

"/stt/google_v2?language=%s&version=%s"
                  % (language, self.version),
                  audio.get_flac_data(),
                  headers=headers)

    if metrics:
      t = timer.stop()
      metrics.timer("mycroft.cerberus.proxy.client.time_s",
t)
      metrics.timer("mycroft.stt.remote.time_s", t)

    if response.status_code == 401:
      raise CerberusAccessDenied()

    try:
      actual_result = response.json()
    except:
      raise UnknownValueError()

    log.info("STT JSON: " + json.dumps(actual_result))
    if show_all:
      return actual_result

    # return the best guess
    if "alternative" not in actual_result:
      raise UnknownValueError()
    alternatives = actual_result["alternative"]
    if len([alt for alt in alternatives if alt.get('confidence')]) >
0:
      # if there is at least one element with confidence,
force it to
      # the front
      alternatives.sort(
        key=lambda e: e.get('confidence', 0.0),
reverse=True)
    for entry in alternatives:
      if "transcript" in entry:
        return entry["transcript"]

    if len(alternatives) > 0:
      log.error(
        "Found %d entries, but none with a transcript." %
len(
          alternatives))

    # no transcriptions available
    raise UnknownValueError()


RECOGNIZER_IMPLS = {
  'google': GoogleRecognizerWrapper,
  'google_proxy': CerberusGoogleProxy,
  'wit': WitRecognizerWrapper,
  'ibm': IBMRecognizerWrapper
}


class RemoteRecognizerWrapperFactory(object):
  @staticmethod
  def wrap_recognizer(recognizer,
impl=config.get('recognizer_impl')):
    if impl not in RECOGNIZER_IMPLS.keys():
      raise NotImplementedError("%s recognizer not
implemented." % impl)

    impl_class = RECOGNIZER_IMPLS.get(impl)
    return impl_class(recognizer)
```

Directory: intelora-core/mycroft/client/speech/word_extractor.py

```
from speech_recognition import AudioData

__author__ = 'jdorleans'


class WordExtractor:
  SILENCE_SECS = 0.1
  PRECISION_RATE = 0.01

  def __init__(self, audio, recognizer, metrics):
    self.audio = audio
    self.recognizer = recognizer
    self.audio_size = len(self.audio.frame_data)
    self.delta = int(self.audio_size / 2)
    self.begin = 0
    self.end = self.audio_size
    self.precision = int(self.audio_size *
self.PRECISION_RATE)
    self.silence_data =
self.create_silence(self.SILENCE_SECS,
                        self.audio.sample_rate,
                        self.audio.sample_width)
    self.metrics = metrics

  def __add(self, is_begin, value):
    if is_begin:
      self.begin += value
    else:
      self.end += value
```

```
    def __calculate_marker(self, is_begin):
        dt = self.delta
        sign = 1 if is_begin else -1

        while dt > self.precision:
            self.__add(is_begin, dt * sign)
            segment = self.audio.frame_data[self.begin:self.end]
            found = self.recognizer.is_recognized(segment,
self.metrics)
            if not found:
                self.__add(is_begin, dt * -sign)
            dt = int(dt / 2)

    def calculate_range(self):
        self.__calculate_marker(False)
        self.__calculate_marker(True)

    @staticmethod
    def create_silence(seconds, sample_rate, sample_width):
        return '\0' * int(seconds * sample_rate * sample_width)

    def get_audio_data_before(self):
        byte_data = self.audio.frame_data[0:self.begin] +
self.silence_data
        return AudioData(byte_data, self.audio.sample_rate,
                         self.audio.sample_width)

    def get_audio_data_after(self):
        byte_data = self.silence_data +
self.audio.frame_data[self.end:
                                            self.audio_size]
        return AudioData(byte_data, self.audio.sample_rate,
                         self.audio.sample_width)
```

Directory: intelora-core/mycroft/client/text/cli.py

```
import sys
from threading import Thread, Lock

from mycroft.messagebus.client.ws import WebsocketClient
from mycroft.messagebus.message import Message
from mycroft.tts import tts_factory
from mycroft.util.log import getLogger

tts = tts_factory.create()
client = None
mutex = Lock()
logger = getLogger("CLIClient")


def handle_speak(event):
    mutex.acquire()

client.emit(Message("recognizer_loop:audio_output_start"))
    try:
        utterance = event.metadata.get('utterance')
        logger.info("Speak: " + utterance)
        tts.execute(utterance)
    finally:
        mutex.release()

client.emit(Message("recognizer_loop:audio_output_end"))


def connect():
    client.run_forever()


def main():
    global client
    client = WebsocketClient()
    if '--quiet' not in sys.argv:
        client.on('speak', handle_speak)
    event_thread = Thread(target=connect)
    event_thread.setDaemon(True)
    event_thread.start()
    try:
        while True:
            print("Input:")
            line = sys.stdin.readline()
            client.emit(
                Message("recognizer_loop:utterance",
                        metadata={'utterances': [line.strip()]}))
    except KeyboardInterrupt, e:
        logger.exception(e)
        event_thread.exit()
        sys.exit()


if __name__ == "__main__":
    main()
```

Directory: intelora-core/mycroft/client/text/main.py

```
import sys
from threading import Thread, Lock

from mycroft.messagebus.client.ws import WebsocketClient
from mycroft.messagebus.message import Message
from mycroft.tts import TTSFactory
from mycroft.util.log import getLogger

tts = TTSFactory.create()
ws = None
mutex = Lock()
logger = getLogger("CLIClient")


def handle_speak(event):
    mutex.acquire()
    ws.emit(Message("recognizer_loop:audio_output_start"))
    try:
        utterance = event.data.get('utterance')
        logger.info("Speak: " + utterance)
        tts.execute(utterance)
    finally:
        mutex.release()

ws.emit(Message("recognizer_loop:audio_output_end"))


def connect():
    ws.run_forever()


def main():
    global ws
```

```
    ws = WebsocketClient()
    tts.init(ws)
    if '--quiet' not in sys.argv:
        ws.on('speak', handle_speak)
    event_thread = Thread(target=connect)
    event_thread.setDaemon(True)
    event_thread.start()
    try:
        while True:
            print("Input:")
            line = sys.stdin.readline()
            ws.emit(
                Message("recognizer_loop:utterance",
                        {'utterances': [line.strip()]}))
    except KeyboardInterrupt, e:
        logger.exception(e)
        event_thread.exit()
        sys.exit()


if __name__ == "__main__":
    main()
```

Directory: intelora-core/mycroft/client/wifisetup/main.py

```
import os
import sys
import threading
import time
import traceback
from SimpleHTTPServer import SimpleHTTPRequestHandler
from SocketServer import TCPServer
from os.path import dirname, realpath
from shutil import copyfile
from subprocess import Popen, PIPE
from threading import Thread
from time import sleep

from pyric import pyw
from wifi import Cell

from mycroft.client.enclosure.api import EnclosureAPI
from mycroft.configuration import ConfigurationManager
from mycroft.messagebus.client.ws import WebsocketClient
from mycroft.messagebus.message import Message
from mycroft.util import connected
from mycroft.util.log import getLogger

__author__ = 'aatchison and penrods'

LOG = getLogger("WiFiClient")

SCRIPT_DIR = dirname(realpath(__file__))


def cli_no_output(*args):
    ''' Invoke a command line and return result '''
    LOG.info("Command: %s" % list(args))
    proc = Popen(args=args, stdout=PIPE, stderr=PIPE)
    stdout, stderr = proc.communicate()
    return {'code': proc.returncode, 'stdout': stdout, 'stderr':
stderr}


def cli(*args):
    ''' Invoke a command line, then log and return result '''
    LOG.info("Command: %s" % list(args))
    proc = Popen(args=args, stdout=PIPE, stderr=PIPE)
    stdout, stderr = proc.communicate()
    result = {'code': proc.returncode, 'stdout': stdout, 'stderr':
stderr}
    LOG.info("Command result: %s" % result)
    return result


def wpa(*args):
    idx = 0
    result = cli('wpa_cli', '-i', *args)
    out = result.get("stdout", "\n")
    if "interface" in out:
        idx = 1
    return str(out.split("\n")[idx])


def sysctrl(*args):
    return cli('systemctl', *args)


class
CaptiveHTTPRequestHandler(SimpleHTTPRequestHandler):
    ''' Serve a single website, 303 redirecting all other
requests to it '''

    def do_HEAD(self):
        LOG.info("do_HEAD being called....")
        if not self.redirect():
            SimpleHTTPRequestHandler.do_HEAD(self)

    def do_GET(self):
        LOG.info("do_GET being called....")
        if not self.redirect():
            SimpleHTTPRequestHandler.do_GET(self)

    def redirect(self):
        try:
            LOG.info("**********************")
            LOG.info("**   HTTP Request   ***")
            LOG.info("**********************")
            LOG.info("Requesting: " + self.path)
            LOG.info("REMOTE_ADDR:" + self.client_address[0])
            LOG.info("SERVER_NAME:" +
self.server.server_address[0])
            LOG.info("SERVER_PORT:" +
str(self.server.server_address[1]))
            LOG.info("SERVER_PROTOCOL:" +
self.request_version)
            LOG.info("HEADERS...")
            LOG.info(self.headers)
            LOG.info("**********************")

            # path = self.translate_path(self.path)
            if "mycroft.ai" in self.headers['host']:
                LOG.info("No redirect")
                return False
            else:
                LOG.info("303 redirect to http://start.mycroft.ai")
                self.send_response(303)
                self.send_header("Location",
"http://start.mycroft.ai")
                self.end_headers()
```

```
            return True
        except:
            tb = traceback.format_exc()
            LOG.info("exception caught")
            LOG.info(tb)
            return False


class WebServer(Thread):
    ''' Web server for devices connected to the temporary
access point '''

    def __init__(self, host, port):
        super(WebServer, self).__init__()
        self.daemon = True
        LOG.info("Creating TCPServer...")
        self.server = TCPServer((host, port),
CaptiveHTTPRequestHandler)
        LOG.info("Created TCPServer")

    def run(self):
        LOG.info("Starting Web Server at %s:%s" %
self.server.server_address)
        LOG.info("Serving from: %s" % os.path.join(SCRIPT_DIR,
'web'))
        os.chdir(os.path.join(SCRIPT_DIR, 'web'))
        self.server.serve_forever()
        LOG.info("Web Server stopped!")


class AccessPoint:
    template = """interface={interface}
bind-interfaces
server={server}
domain-needed
bogus-priv
dhcp-range={dhcp_range_start}, {dhcp_range_end}, 12h
address=/#/{server}
"""

    def __init__(self, wiface):
        self.wiface = wiface
        self.iface = 'p2p-wlan0-0'
        self.subnet = '172.24.1'
        self.ip = self.subnet + '.1'
        self.ip_start = self.subnet + '.50'
        self.ip_end = self.subnet + '.150'
        self.password = None

    def up(self):
        try:
            card = pyw.getcard(self.iface)
        except:
            wpa(self.wiface, 'p2p_group_add', 'persistent=0')
            self.iface = self.get_iface()
            self.password = wpa(self.iface, 'p2p_get_passphrase')
            card = pyw.getcard(self.iface)
        pyw.inetset(card, self.ip)
        copyfile('/etc/dnsmasq.conf', '/tmp/dnsmasq-bk.conf')
        self.save()
        sysctrl('restart', 'dnsmasq.service')

    def get_iface(self):
        for iface in pyw.winterfaces():
            if "p2p" in iface:
                return iface

    def down(self):
        sysctrl('stop', 'dnsmasq.service')
        sysctrl('disable', 'dnsmasq.service')
        wpa(self.wiface, 'p2p_group_remove', self.iface)
        copyfile('/tmp/dnsmasq-bk.conf', '/etc/dnsmasq.conf')

    def save(self):
        data = {
            "interface": self.iface,
            "server": self.ip,
            "dhcp_range_start": self.ip_start,
            "dhcp_range_end": self.ip_end
        }
        try:
            LOG.info("Writing to: /etc/dnsmasq.conf")
            with open('/etc/dnsmasq.conf', 'w') as f:
                f.write(self.template.format(**data))
        except Exception as e:
            LOG.error("Fail to write: /etc/dnsmasq.conf")
            raise e


class WiFi:
    def __init__(self):
        self.iface = pyw.winterfaces()[0]
        self.ap = AccessPoint(self.iface)
        self.server = None
        self.ws = WebsocketClient()
        ConfigurationManager.init(self.ws)
        self.enclosure = EnclosureAPI(self.ws)
        self.init_events()
        self.conn_monitor = None
        self.conn_monitor_stop = threading.Event()

    def init_events(self):
        '''
            Register handlers for various websocket events used
            to communicate with outside systems.
        '''

        # This event is generated by an outside mechanism.  On
a
        # Holmes unit this comes from the Enclosure's menu
item
        # being selected.
        self.ws.on('mycroft.wifi.start', self.start)

        # These events are generated by Javascript in the
captive
        # portal.
        self.ws.on('mycroft.wifi.stop', self.stop)
        self.ws.on('mycroft.wifi.scan', self.scan)
        self.ws.on('mycroft.wifi.connect', self.connect)

    def start(self, event=None):
        '''
          Fire up the MYCROFT access point for the user to
connect to
          with a phone or computer.
        '''
        LOG.info("Starting access point...")

        # Fire up our access point
        self.ap.up()
        if not self.server:
```

```
            LOG.info("Creating web server...")
            self.server = WebServer(self.ap.ip, 80)
            LOG.info("Starting web server...")
            self.server.start()
            LOG.info("Created web server.")

        LOG.info("Access point started!\n%s" %
self.ap.__dict__)
        self._start_connection_monitor()

    def _connection_prompt(self, prefix):
        # let the user know to connect to it...
        passwordSpelled = ",  ".join(self.ap.password)
        self._speak_and_show(
            prefix + " Use your mobile device or computer to "
                "connect to the wifi network "
                "'MYCROFT';  Then enter the uppercase "
                "password " + passwordSpelled,
            self.ap.password)

    def _speak_and_show(self, speak, show):
        ''' Communicate with the user throughout the process '''
        self.ws.emit(Message("speak", {'utterance': speak}))
        if show is None:
            return

        # TODO: This sleep should not be necessary, but
without it the
        #       text to be displayed by enclosure.mouth_text()
gets
        #       wiped out immediately when the utterance above
is
        #       begins processing.
        #       Remove the sleep once this behavior is corrected.
        sleep(0.25)
        self.enclosure.mouth_text(show)

    def _start_connection_monitor(self):
        LOG.info("Starting monitor thread...\n")
        if self.conn_monitor is not None:
            LOG.info("Killing old thread...\n")
            self.conn_monitor_stop.set()
            self.conn_monitor_stop.wait()

        self.conn_monitor = threading.Thread(
            target=self._do_connection_monitor,
            args={})
        self.conn_monitor.daemon = True
        self.conn_monitor.start()
        LOG.info("Monitor thread setup complete.\n")

    def _stop_connection_monitor(self):
        ''' Set flag that will let monitoring thread close '''
        self.conn_monitor_stop.set()

    def _do_connection_monitor(self):
        LOG.info("Invoked monitor thread...\n")
        mtimeLast =
os.path.getmtime('/var/lib/misc/dnsmasq.leases')
        bHasConnected = False
        cARPFailures = 0
        timeStarted = time.time()
        timeLastAnnounced = 0  # force first announcement to
now
        self.conn_monitor_stop.clear()
        while not self.conn_monitor_stop.isSet():
            # do our monitoring...
            mtime =
os.path.getmtime('/var/lib/misc/dnsmasq.leases')
            if mtimeLast != mtime:
                # Something changed in the dnsmasq lease file -
                # presumably a (re)new lease
                bHasConnected = True
                cARPFailures = 0
                mtimeLast = mtime
                timeStarted = time.time()  # reset start time after
connection
                timeLastAnnounced = time.time() - 45  # announce
how to connect

            if time.time() - timeStarted > 60 * 5:
                # After 5 minutes, shut down the access point
                LOG.info("Auto-shutdown of access point after 5
minutes")
                self.stop()
                continue

            if time.time() - timeLastAnnounced >= 45:
                if bHasConnected:
                    self._speak_and_show(
                        "Now you can open your browser and go to
start dot "
                        "mycroft dot A I, then follow the instructions
given "
                        " there",
                        "start.mycroft.ai")
                else:
                    self._connection_prompt("Allow me to walk you
through the "
                                            " wifi setup process; ")
                timeLastAnnounced = time.time()

            if bHasConnected:
                # Flush the ARP entries associated with our access
point
                # This will require all network hardware to re-
register
                # with the ARP tables if still present.
                if cARPFailures == 0:
                    res = cli_no_output('ip', '-s', '-s', 'neigh', 'flush',
                                        self.ap.subnet + '.0/24')
                    # Give ARP system time to re-register hardware
                    sleep(5)

                # now look at the hardware that has responded, if
no entry
                # shows up on our access point after 2*5=10
seconds, the user
                # has disconnected
                if not self._is_ARP_filled():
                    cARPFailures += 1
                    if cARPFailures > 2:
                        self._connection_prompt("Connection lost,")
                        bHasConnected = False
                else:
                    cARPFailures = 0
            sleep(5)  # wait a bit to prevent thread from hogging
CPU

        LOG.info("Exiting monitor thread...\n")
        self.conn_monitor_stop.clear()
```

```
    def _is_ARP_filled(self):
        res = cli_no_output('/usr/sbin/arp', '-n')
        out = str(res.get("stdout"))
        if out:
            # Parse output, skipping header
            for o in out.split("\n")[1:]:
                if o[0:len(self.ap.subnet)] == self.ap.subnet:
                    if "(incomplete)" in o:
                        # ping the IP to get the ARP table entry
reloaded
                        ip_disconnected = o.split(" ")[0]
                        cli_no_output('/bin/ping', '-c', '1', '-W', 3,
                                      ip_disconnected)
                    else:
                        return True  # something on subnet is
connected!
        return False

    def scan(self, event=None):
        LOG.info("Scanning wifi connections...")
        networks = {}
        status = self.get_status()

        for cell in Cell.all(self.iface):
            update = True
            ssid = cell.ssid
            quality = self.get_quality(cell.quality)

            # If there are duplicate network IDs (e.g. repeaters)
only
            # report the strongest signal
            if networks.__contains__(ssid):
                update = networks.get(ssid).get("quality") < quality
            if update and ssid:
                networks[ssid] = {
                    'quality': quality,
                    'encrypted': cell.encrypted,
                    'connected': self.is_connected(ssid, status)
                }
        self.ws.emit(Message("mycroft.wifi.scanned",
                             {'networks': networks}))
        LOG.info("Wifi connections scanned!\n%s" % networks)

    @staticmethod
    def get_quality(quality):
        values = quality.split("/")
        return float(values[0]) / float(values[1])

    def connect(self, event=None):
        if event and event.data:
            ssid = event.data.get("ssid")
            connected = self.is_connected(ssid)

            if connected:
                LOG.warn("Mycroft is already connected to %s" %
ssid)
            else:
                self.disconnect()
                LOG.info("Connecting to: %s" % ssid)
                nid = wpa(self.iface, 'add_network')
                wpa(self.iface, 'set_network', nid, 'ssid', '"' + ssid +
'"')

                if event.data.__contains__("pass"):
                    psk = '"' + event.data.get("pass") + '"'
                    wpa(self.iface, 'set_network', nid, 'psk', psk)
                else:
                    wpa(self.iface, 'set_network', nid, 'key_mgmt',
'NONE')

                wpa(self.iface, 'enable', nid)
                connected = self.get_connected(ssid)
                if connected:
                    wpa(self.iface, 'save_config')

            self.ws.emit(Message("mycroft.wifi.connected",
                                 {'connected': connected}))
            LOG.info("Connection status for %s = %s" % (ssid,
connected))

            if connected:
                self.ws.emit(Message("speak", {
                    'utterance': "Thank you, I'm now connected to
the "
                                 "internet and ready for use"}))
                # TODO: emit something that triggers a pairing
check

    def disconnect(self):
        status = self.get_status()
        nid = status.get("id")
        if nid:
            ssid = status.get("ssid")
            wpa(self.iface, 'disable', nid)
            LOG.info("Disconnecting %s id: %s" % (ssid, nid))

    def get_status(self):
        res = cli('wpa_cli', '-i', self.iface, 'status')
        out = str(res.get("stdout"))
        if out:
            return dict(o.split("=") for o in out.split("\n")[:-1])
        return {}

    def get_connected(self, ssid, retry=5):
        connected = self.is_connected(ssid)
        while not connected and retry > 0:
            sleep(2)
            retry -= 1
            connected = self.is_connected(ssid)
        return connected

    def is_connected(self, ssid, status=None):
        status = status or self.get_status()
        state = status.get("wpa_state")
        return status.get("ssid") == ssid and state ==
"COMPLETED"

    def stop(self, event=None):
        LOG.info("Stopping access point...")
        self._stop_connection_monitor()
        self.ap.down()
        if self.server:
            self.server.server.shutdown()
            self.server.server.server_close()
            self.server.join()
            self.server = None
        LOG.info("Access point stopped!")

    def _do_net_check(self):
        # give system 5 seconds to resolve network or get
plugged in
```

```
        sleep(5)

        LOG.info("Checking internet connection again")
        if not connected() and self.conn_monitor is None:
            # TODO: Enclosure/localization
            self._speak_and_show(
                "This device is not connected to the Internet. Either
plug "
                "in a network cable or hold the button on top for
two "
                "seconds, then select wifi from the menu", None)

    def run(self):
        try:
            # When the system first boots up, check for a valid
internet
            # connection.
            LOG.info("Checking internet connection")
            if not connected():
                LOG.info("No connection initially, waiting 20...")
                self.net_check = threading.Thread(
                    target=self._do_net_check,
                    args={})
                self.net_check.daemon = True
                self.net_check.start()
            else:
                LOG.info("Connection found!")

            self.ws.run_forever()
        except Exception as e:
            LOG.error("Error: {0}".format(e))
            self.stop()


def main():
    wifi = WiFi()
    try:
        wifi.run()
    except Exception as e:
        print (e)
    finally:
        sys.exit()


if __name__ == "__main__":
    main()
```

Directory: intelora-core/mycroft/client/wifisetup/web/css/style.css

```
* {
    font-family: sans-serif;
    user-select: none;
    -moz-user-select: none;
    -webkit-user-select: text;
    -ms-user-select: none;
}

/* Prevent ridiculously long network names from screwing
up the layout
   by forcing them to wrap. */
.ssid {
    max-width: 280px;
}

/* A bug in Safari (as of 10-10-2016) breaks password input
when
   user-select:none is used.  This prevents that bug by
allowing
   text selection for all input fields (which is probably
expected
   by users anyway). */
input {
    user-select: text;
    -moz-user-select: text;
    -webkit-user-select: text;
    -ms-user-select: text;
}

.panel {
    border-radius: 15px;
    box-shadow: 0 0 10px #AAA;
    max-width: 500px;
    min-width: 300px;
}

.panel .title {
    background: #2b3344;
    border-top-left-radius: 15px;
    border-top-right-radius: 15px;
    padding: 30px 20px;
    text-align: center;
    color: #FFF;
    font-weight: bold;
    font-size: 20px;
}

.panel .body {
    min-height: 520px;
    display: table;
    list-style: none;
    padding: 0;
    margin: 0;
    width: 100%;
}

.panel .body li {
    width: 100%;
    display: table;
    border-bottom: 1px solid #DDD;
    padding: 10px;
    box-sizing: border-box;
    cursor: pointer;
    height: 61px;
    overflow: hidden;
}

.panel .body li div {
    width: 100%;
    height: 40px;
    transition: 500ms;
    display: none;
}

.panel .body li div.show {
    transition: 500ms;
    display: table;
}

.panel .body li span {
```

```
    float: left;
    font-size: 18px;
    margin: 9px 0;
    width: calc(100% - 60px);
    text-overflow: ellipsis;
    word-wrap: break-word;
}

.panel .body li img {
    width: 40px;
    float: right;
}

.panel .body li img.lock {
    width: 11px;
    position: relative;
    right: 0;
    bottom: 0;
    top: 24px;
    left: 40px;
}

.panel .body li .connect-item img, .panel .body li .error-item
img {
    width: 20px;
    height: 20px;
    padding: 10px;
}

.panel .body li .connect-item label {
    color: #488fe2;
    padding: 9.5px 10px 9.5px 0;
    display: table;
    float: left;
    font-size: 18px;
    width: 80px;
    text-align: left;
}

.panel .body li .connect-item label.public {
    width: calc(100% - 60px);
}

.panel .body li .connect-item input {
    padding: 5px;
    height: 26px;
    border: 1px solid #DDD;
    width: calc(100% - 160px);
    font-size: 16px;
}

.panel .body li .error-item span {
    color: #ff6565;
    font-weight: bold;
    padding: 2px;
    display: table;
    float: left;
    font-size: 16px;
    text-align: left;
    width: calc(100% - 60px);
}

#success .title {
    background: #b8e986;
    color: #2b3344;
}

#success span img {
    position: relative;
    top: 4px;
}

#success small {
    text-align: left;
    width: 80%;
    margin: 35px auto;
    font-size: 18px;
    display: table;
    color: #777;
}

div.connected span {
    padding: 0 !important;
    margin: 0 !important;
}

div.connected span:not(.connected) {
    color: #488fe2;
}

div.connected .lock {
    top: 3px !important;
}

span.connected {
    display: table;
    margin: 0 !important;
    padding: 0 !important;
    font-size: 14px !important;
    color: #777;
}

.backButton {
    color: #4b90e2;
    font-weight: bold;
    display: table;
    width: 100%;
    padding: 10px;
    box-sizing: border-box;
    cursor: pointer;
}

.backButton img {
    width: 15px;
    float: left;
    margin: 2.5px 0;
}

.backButton span {
    float: left;
    padding: 0 3px;
    display: table;
    font-size: 20px;
}

.message {
    width: 100%;
    display: table;
    text-align: center;
    margin: 30px 0 40px 0;
    box-sizing: border-box;
    padding: 10px 20px;
```

```
}

.passForm {
  display: table;
  width: 280px;
  box-sizing: border-box;
  margin: 0 auto 50px auto;
}

.passForm label {
  color: #444;
}

.passForm label input {
  width: 100%;
  display: table;
  height: 40px;
  border: 1px solid #444;
  border-radius: 5px;
  margin: 8px 0;
  box-sizing: border-box;
  padding: 10px;
}

.button {
  margin: 0 auto;
  display: table;
  background: #4a90e2;
  border: none;
  padding: 10px;
  border-radius: 5px;
  color: #FFF;
  cursor: pointer;
  text-decoration: none;
}

#centered {
  margin-right:auto;
  margin-left:auto;
}
#footer {
  position:fixed;
  bottom:10px;
  width:100%;
}
#cancelBtn {
  background: white;
  z-index: 999;
  color: #ff6666;
  padding: 7px;   /* 10px of .button -3px to account for the
thick border */
  border: 3px solid #ff6666;
}

.alert {
  width: 100%;
  text-align: center;
  box-sizing: border-box;
  margin-bottom: 10px;
  color: #ff9595;
  font-weight: bold;
  height: 0;
  overflow: hidden;
  transition: 500ms;
}

.alert.show {
  border: 1px solid #ff9595;
  padding: 10px;
  height: 40px;
  transition: 500ms;
}

.hide {
  display: none;
}

#home span, #success span {
  color: #4a4a4a;
  font-weight: bold;
  font-size: 22px;
  text-align: left;
  width: 80%;
  margin: 80px auto 0 auto;
  display: table;
}

#home img {
  width: 100%;
  margin: 60px 0;
}

@media (min-width: 500px) {
  .panel {
    margin: 0 auto;
  }
}

/* loading */
.loader {
  color: #4a90e2;
  font-size: 20px;
  margin: 150px auto;
  width: 1em;
  height: 1em;
  border-radius: 50%;
  position: relative;
  text-indent: -9999em;
  -webkit-animation: load4 1.3s infinite linear;
  animation: load4 1.3s infinite linear;
  -webkit-transform: translateZ(0);
  -ms-transform: translateZ(0);
  transform: translateZ(0);
}

@-webkit-keyframes load4 {
  0%,
  100% {
    box-shadow: 0 -3em 0 0.2em, 2em -2em 0 0em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 -1em, -
3em 0 0 -1em, -2em -2em 0 0;
  }
  12.5% {
    box-shadow: 0 -3em 0 0, 2em -2em 0 0.2em, 3em 0 0 0,
2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 -1em, -3em 0 0
-1em, -2em -2em 0 -1em;
  }
  25% {
    box-shadow: 0 -3em 0 -0.5em, 2em -2em 0 0, 3em 0 0
0.2em, 2em 2em 0 0, 0 3em 0 -1em, -2em 2em 0 -1em, -3em
0 0 -1em, -2em -2em 0 -1em;
  }
```

```
  37.5% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em
0em 0 0, 2em 2em 0 0.2em, 0 3em 0 0em, -2em 2em 0 -
1em, -3em 0em 0 -1em, -2em -2em 0 -1em;
  }
  50% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 0em, 0 3em 0 0.2em, -2em 2em 0 0, -3em
0em 0 -1em, -2em -2em 0 -1em;
  }
  62.5% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 0, -2em 2em 0 0.2em, -3em
0 0 0, -2em -2em 0 -1em;
  }
  75% {
    box-shadow: 0em -3em 0 -1em, 2em -2em 0 -1em, 3em
0em 0 -1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 0,
-3em 0em 0 0.2em, -2em -2em 0 0;
  }
  87.5% {
    box-shadow: 0em -3em 0 0, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 0, -3em
0em 0 0, -2em -2em 0 0.2em;
  }
}

@keyframes load4 {
  0%,
  100% {
    box-shadow: 0 -3em 0 0.2em, 2em -2em 0 0em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 -1em, -
3em 0 0 -1em, -2em -2em 0 0;
  }
  12.5% {
    box-shadow: 0 -3em 0 0, 2em -2em 0 0.2em, 3em 0 0 0,
2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 -1em, -3em 0 0
-1em, -2em -2em 0 -1em;
  }
  25% {
    box-shadow: 0 -3em 0 -0.5em, 2em -2em 0 0, 3em 0 0
0.2em, 2em 2em 0 0, 0 3em 0 -1em, -2em 2em 0 -1em, -3em
0 0 -1em, -2em -2em 0 -1em;
  }
  37.5% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em
0em 0 0, 2em 2em 0 0.2em, 0 3em 0 0em, -2em 2em 0 -
1em, -3em 0em 0 -1em, -2em -2em 0 -1em;
  }
  50% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 0em, 0 3em 0 0.2em, -2em 2em 0 0, -3em
0em 0 -1em, -2em -2em 0 -1em;
  }
  62.5% {
    box-shadow: 0 -3em 0 -1em, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 0, -2em 2em 0 0.2em, -3em
0 0 0, -2em -2em 0 -1em;
  }
  75% {
    box-shadow: 0em -3em 0 -1em, 2em -2em 0 -1em, 3em
0em 0 -1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 0,
-3em 0em 0 0.2em, -2em -2em 0 0;
  }
  87.5% {
    box-shadow: 0em -3em 0 0, 2em -2em 0 -1em, 3em 0 0
-1em, 2em 2em 0 -1em, 0 3em 0 -1em, -2em 2em 0 0, -3em
0em 0 0, -2em -2em 0 0.2em;
  }
}
```

Directory: intelora-core/mycroft/client/wifisetup/web/js/main.js

```
function getImagePath(strength) {
  if (strength > 0.8) {
    return "img/wifi_4.png";
  } else if (strength > 0.6) {
    return "img/wifi_3.png";
  } else if (strength > 0.4) {
    return "img/wifi_2.png";
  } else if (strength > 0.2) {
    return "img/wifi_1.png";
  } else {
    return "img/wifi_0.png";
  }
}

function showPanel(id) {
  var panels = document.querySelectorAll(".panel");

  for (var i=0; i < panels.length; i++)
    panels[i].classList.add("hide");

  document.querySelector("#" +
id).classList.remove("hide");
}

var WifiSetup = {

    selectedNetword: null,

    setListeners: function () {
      WS.addMessageListener("mycroft.wifi.connected",
this.onConnected.bind(this));
      WS.addMessageListener("mycroft.wifi.scanned",
this.onScanned.bind(this));
    },

    onConnected: function (data) {
      if (data.connected) {
        // NOTE:  Once we send the "mycroft.wifi.stop", the
unit will
        // be shutting down the wifi access point.  So the
device
        // hosting the browser session is probably being
disconnected
        // and hopefully automatically reconnecting to the
internet.
        //
        // Until the reconnect happens, the user cannot
actually
        // follow the link to http://home.mycroft.ai to
register
        // their device.  That is part of why we are doing
this 2 sec
        // delay.
        //
        WS.send("mycroft.wifi.stop");
        WS.close();
```

```
            setTimeout(function () {
                var btnCancel =
document.querySelector("#cancelBtn");
                btnCancel.classList.add("hide");

                showPanel("success");
                startPing();
            }, 2000);
        } else {
            showPanel("list-panel");
            this.renderErrorItem(this.selectedNetword.el);
        }
    },

    onScanned: function (data) {
        var networks = data.networks,
            fragment = document.createDocumentFragment(),
            list = document.querySelector("#list"),
            item = null,
            li = null;

        showPanel("list-panel");

        Object.keys(networks).sort(function (a, b) {
            if (networks[a].quality < networks[b].quality) {
                return 1;
            }
            return 0;
        }).forEach(function (network) {
            li = document.createElement("li");
            networks[network].ssid = network;
            networks[network].el = li;
            item = this.renderListItem(networks[network]);
            li.appendChild(item);
            fragment.appendChild(li);
        }.bind(this));

        list.innerHTML = null;
        list.appendChild(fragment);
    },

    renderListItem: function (network) {
        var listItem = document.createElement("div"),
            span = document.createElement("span"),
            imgSignal = document.createElement("img");
        listItem.className = "list-item show";
        span.textContent = network.ssid;
        span.className = "ssid";
        imgSignal.src = getImagePath(network.quality);
        imgSignal.className = "wifi";
        listItem.appendChild(span);
        listItem.appendChild(imgSignal);
        if (network.connected) {
            var connected = document.createElement("span");
            connected.className = "connected";
            connected.textContent = "Connected";
            listItem.classList.add("connected");
            listItem.appendChild(connected);
        } else {
            listItem.addEventListener("click",
this.clickNetwork.bind(this, network));
        }
        if (network.encrypted) {
            var imgLock = document.createElement("img");
            imgLock.src = "img/lock.png";
            imgLock.className = "lock";
            listItem.appendChild(imgLock);
        }
        return listItem;
    },

    ItemDefaultState: function () {
        var li = document.querySelector(".list-
item:not(.show)");
        if (!li) {
            return;
        }
        var divs = li.parentNode.childNodes;

        Object.keys(divs).forEach(function (div) {
            divs[div].classList.remove("show");
            if (divs[div].classList.contains("list-item")) {
                divs[div].classList.add("show");
            }
        });
    },

    renderConnectItem: function (li) {
        var connect = li.querySelector(".connect-item");
        if (!connect) {
            connect = document.createElement("div");
            var imgArrow = document.createElement("img");
            var label = document.createElement("label");
            connect.className = "connect-item";
            imgArrow.src = "img/next.png";
            imgArrow.addEventListener("click",
this.clickConnect.bind(this));
            connect.appendChild(label);
            if (this.selectedNetword.encrypted) {
                connect.passwordInput =
document.createElement("input");
                label.textContent = "Password: ";
                connect.passwordInput.type = "password";
                connect.appendChild(connect.passwordInput);
            } else {
                label.className = "public";
                label.textContent = this.selectedNetword.ssid;
            }
            connect.appendChild(imgArrow);

            li.appendChild(connect);
                }
        li.querySelector(".list-
item").classList.remove("show");
        connect.classList.add("show");
        if ('passwordInput' in connect)
                            connect.passwordInput.focus();
    },

    renderErrorItem: function (li) {
        var error = li.querySelector(".error-item");
        if (error) {
            li.querySelector(".connect-
item").classList.remove("show");
            error.classList.add("show");
            return;
        }
        error = document.createElement("div");
        var imgClose = document.createElement("img");
        error.classList.add("error-item");
        imgClose.src = "img/error.png";
        var message = document.createElement("span");
```

```
        message.textContent = "Try again or connect to a
diferent wifi.";
        error.appendChild(message);
        error.appendChild(imgClose);
        li.appendChild(error);
        li.querySelector(".connect-
item").classList.remove("show");
        error.classList.add("show");
        error.addEventListener("click", this.ItemDefaultState);
    },

    clickNetwork: function (network) {
      this.selectedNetword = network;
      this.ItemDefaultState();
      this.renderConnectItem(network.el);
    }
    ,

    sendScan: function () {
      showPanel("loading");

document.querySelector("#cancelBtn").classList.remove("hi
de");
      WS.send("mycroft.wifi.scan");
    }
    ,

    /***
     * @param data is a object with ssid and pass
     */
    sendConnect: function (data) {
      WS.send("mycroft.wifi.connect", data);
    }
    ,

    clickConnect: function () {
      showPanel("connecting");
      var network = {
        ssid: this.selectedNetword.ssid
      };
      if (this.selectedNetword.encrypted) {
        var pass =
this.selectedNetword.el.querySelector("input");
        network.pass = pass.value;
      }
      this.sendConnect(network);
    }
    ,

    cancelSetup: function () {
      WS.send("mycroft.wifi.stop");
      WS.close();
    }
    ,

    init: function () {
      this.setListeners();
      showPanel("home");

document.querySelector("#connectBtn").addEventListener("
click", this.sendScan);

document.querySelector("#registerBtn").addEventListener("
click", function () {
        setTimeout(function() {
          location.href="https://home.mycroft.ai";
        }, 2000);
      });

document.querySelector("#cancelBtn").addEventListener("cl
ick", this.cancelSetup);
    }
  };

function startPing() {
  ping("home.mycroft.ai",
    function(status,e) {
      if (status == 'responded') {
        // Un-hide the register button once we detect an
        // active internet connection.

document.querySelector("#registerBtn").classList.remove("h
ide");
      }
      else
        setTimeout(function() { startPing(); }, 1000);
    });
}

function ping(domain, callback) {
  if (!this.inUse) {
    this.status = 'unchecked';
    this.inUse = true;
    this.callback = callback;
    this.ip = domain;
    var _that = this;
    this.img = new Image();
    this.img.onload = function () {
      _that.inUse = false;
      _that.callback('responded');

    };
    this.img.onerror = function (e) {
      if (_that.inUse) {
        _that.inUse = false;
        _that.callback('responded', e);
      }

    };
    this.start = new Date().getTime();
    this.img.src = "http://" + domain;
    this.timer = setTimeout(function () {
      if (_that.inUse) {
        _that.inUse = false;
        _that.callback('timeout');
      }
    }, 1500);
  }
}

window.addEventListener("load", function () {
  WS.connect();
  WS.setOnOpenListener(function () {
    WifiSetup.init();
  });
});
```

Directory: intelora-core/mycroft/client/wifisetup/web/js/Config.js

```
var Config = {
  wsUrl: "ws://172.24.1.1:8181/core"
};
```

Directory: intelora-core/mycroft/client/wifisetup/web/js/WS.js

```
var WS = {
  ws: null,
  wsConnected: false,
  listeners: {},
  onOpenListeners: [],

  connect: function () {
    this.ws = new WebSocket(Config.wsUrl);
    this.setWSListeners();
  },

  setWSListeners: function () {
    this.ws.onmessage = this.onMessage.bind(this);
    this.ws.onopen = this.onOpen.bind(this);
  },

  setOnOpenListener: function (cb) {
    this.onOpenListeners.push(cb);
  },

  onMessage: function (evt) {
    var msg = JSON.parse(evt.data);
    if (this.listeners[msg.type]) {
      this.listeners[msg.type].forEach(function (cb) {
        cb(msg.data);
      });
    }
  },

  onOpen: function () {
    this.wsConnected = true;
    this.onOpenListeners.forEach(function (cb) {
      cb();
    });
  },

  send: function (type, data) {
    this.ws.send(JSON.stringify({
      type: type,
      data: data
    }));
  },

  close: function () {
    this.ws.close();
    this.wsConnected = false;
    this.ws = null;
  },

  addMessageListener: function (type, callback) {
    this.listeners[type] = this.listeners[type] || [];
    this.listeners[type].push(callback);
  }

};
```

Directory: intelora-core/mycroft/client/wifisetup/web/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Wifi Setup</title>
  <link rel="stylesheet" href="css/style.css">
  <meta name="viewport" content="width=device-width,
initial-scale=1">
  <script type="application/javascript"
src="js/Config.js"></script>
  <script type="application/javascript"
src="js/WS.js"></script>
  <script type="application/javascript"
src="js/main.js"></script>
</head>
<body>

<div id="home" class="panel hide">
  <div class="title">HI, MYCROFT HERE</div>
  <div class="body">
    <span>This process is for connecting me to the internet
through your wifi.</span>
    <img src="img/connect.png"/>
    <a id="connectBtn" class="button">LET'S CONNECT</a>
  </div>
</div>

<div id="list-panel" class="panel hide">
  <div class="title">CHOOSE YOUR USUAL WIFI</div>
  <ul id="list" class="body"></ul>
</div>

<div id="loading" class="panel">
  <div class="title">LOADING</div>
  <div class="body">
    <div class="loader"></div>
  </div>
</div>

<div id="connecting" class="panel hide">
  <div class="title">CONNECTING</div>
  <div class="body">
    <div class="loader"></div>
  </div>
</div>

<div id="success" class="panel hide">
  <div class="title">CONNECTED</div>
  <div class="body">
    <span>I'm connected to the internet now. Feels goood
<img src="img/love.png"/></span>
    <small>If I am registered with your account, I'm ready
to roll.  If not, go ahead and register me at
home.mycroft.ai.</small>
    <a class="button hide" id="registerBtn">REGISTER
ME</a>
  </div>
</div>

<p class="centered">
  <a class="button" id="cancelBtn">CANCEL SETUP</a>
</p>

</body>
</html>
```

Directory: intelora-core/scripts/start.sh

```
#!/usr/bin/env bash
TOP=$(cd $(dirname $0) && cd .. && pwd -L)
VIRTUALENV_ROOT=${VIRTUALENV_ROOT:-
"${HOME}/.virtualenvs/mycroft"}

case $1 in
            "service")
SCRIPT=${TOP}/mycroft/messagebus/service/main.py ;;
            "skills") SCRIPT=${TOP}/mycroft/skills/main.py ;;
            "skill_container")
SCRIPT=${TOP}/mycroft/skills/container.py ;;
            "voice")
SCRIPT=${TOP}/mycroft/client/speech/main.py ;;
            "cli") SCRIPT=${TOP}/mycroft/client/text/main.py
;;
            "audiotest")
SCRIPT=${TOP}/mycroft/util/audio_test.py ;;
            "collector")
SCRIPT=${TOP}/mycroft_data_collection/cli.py ;;
            "unittest") SCRIPT=${TOP}/test/main.py ;;
            "audioaccuracytest") SCRIPT=${TOP}/test/audio-
accuracy-test/audio_accuracy_test.py ;;
            "sdkdoc")
SCRIPT=${TOP}/doc/generate_sdk_docs.py ;;
  "enclosure")
SCRIPT=${TOP}/mycroft/client/enclosure/main.py ;;
  "wifi") SCRIPT=${TOP}/mycroft/client/wifisetup/main.py ;;
            *) echo "Usage: start.sh [service | skills |
skill_container | voice | cli | audiotest | collector | unittest |
enclosure | sdkdoc | wifi]"; exit ;;
esac

echo "Starting $@"

shift

source ${VIRTUALENV_ROOT}/bin/activate
PYTHONPATH=${TOP} python ${SCRIPT} $@
```

Directory: intelora-core/scripts/install-mimic.sh

```
#!/usr/bin/env bash
# exit on any error
set -Ee

MIMIC_DIR=mimic
CORES=$(nproc)

# download and install mimic
if [ ! -d ${MIMIC_DIR} ]; then
  git clone https://github.com/MycroftAI/mimic.git
  cd ${MIMIC_DIR}
  ./configure --with-audio=alsa
  make #-j$CORES
else
  # ensure mimic is up to date
  cd ${MIMIC_DIR}
  git pull
  make clean
  ./configure --with-audio=alsa
  make #-j$CORES
fi
```

Directory: intelora-core/scripts/pocketsphinx.sh

```
#!/usr/bin/env bash
# exit on any error
set -Ee

#TOP="."


function enable_local {
 sed -i -- 's/from pocketsphinx.pocketsphinx import
Decoder/from pocketsphinx import Decoder/g'
mycroft/client/speech/local_recognizer.py
}

function disable_local {
 sed -i -- 's/from pocketsphinx import Decoder/from
pocketsphinx.pocketsphinx import Decoder/g'
mycroft/client/speech/local_recognizer.py

}
function install_pocketsphinx {
 # clone pocketsphinx-python at HEAD (fix to a constant
version later)
 if [ ! -d ${TOP}/pocketsphinx-python ]; then
  # build sphinxbase and pocketsphinx if we haven't already
  git clone --recursive
https://github.com/cmusphinx/pocketsphinx-python
  pushd ./pocketsphinx-python/sphinxbase
  ./autogen.sh
  ./configure
  make -j$CORES
  popd
  pushd ./pocketsphinx-python/pocketsphinx
  ./autogen.sh
  ./configure
  make -j$CORES
  popd
 fi

 # build and install pocketsphinx python bindings
 cd ${TOP}/pocketsphinx-python
 python setup.py install

}

if [ "$1" = "-q" ]; then
 enable_local
 install_pocketsphinx
 exit 0
fi

echo "This script will checkout, compile, and install
pocketsphinx locally if the debian package python-
pocketsphinx is not available"

PS3='Please enter your choice: '
options=("Enable local checkout, compile and install"
"Disable local checkout and exit" "Do nothing and quit")
select opt in "${options[@]}"
do
  case $opt in
```

```
      "Enable local checkout, compile and install")
        echo "you chose choice 1"
        enable_local
        install_pocketsphinx
        ;;
      "Disable local checkout and exit")
        echo "you chose choice 2"
        disable_local
        exit 0
        ;;
      "Do nothing and quit")
        echo "you chose choice 3"
        exit 0
        ;;
            *) echo invalid option;;
  esac
done
```

Directory: intelora-core/scripts/install-pygtk.sh

```
#!/usr/bin/env bash
# Ensure we're in a virtualenv.
if [ "$VIRTUAL_ENV" == "" ]
then
  echo "ERROR: not in a virtual environment."
  exit -1
fi

# Setup variables.
CACHE="/tmp/install-pygtk-$$"
CORES=$(nproc)

# Make temp directory.
mkdir -p $CACHE

# Test for py2cairo.
echo -e "\E[1m * Checking for cairo...\E[0m"
python -c "
try: import cairo; raise SystemExit(0)
except ImportError: raise SystemExit(-1)"

if [ $? == 255 ]
then
  echo -e "\E[1m * Installing cairo...\E[0m"
  # Fetch, build, and install py2cairo.
  (  cd $CACHE
     curl 'https://www.cairographics.org/releases/py2cairo-
1.10.0.tar.bz2' > "py2cairo.tar.bz2"
     tar -xvf py2cairo.tar.bz2
     (  cd py2cairo*
        autoreconf -ivf
        ./configure --prefix=$VIRTUAL_ENV --disable-
dependency-tracking
        make -j$CORES
        make install
     )
  )
fi

# Test for gobject.
echo -e "\E[1m * Checking for gobject...\E[0m"
python -c "
try: import gobject; raise SystemExit(0)
except ImportError: raise SystemExit(-1)"

if [ $? == 255 ]
then
  echo -e "\E[1m * Installing gobject...\E[0m"
  # Fetch, build, and install gobject.
  (  cd $CACHE
     curl
'http://ftp.gnome.org/pub/GNOME/sources/pygobject/2.28
/pygobject-2.28.6.tar.bz2' > 'pygobject.tar.bz2'
     tar -xvf pygobject.tar.bz2
     (  cd pygobject*
        ./configure --prefix=$VIRTUAL_ENV --disable-
introspection
        make -j$CORES
        make install
     )
  )
fi

# Test for gtk.
echo -e "\E[1m * Checking for gtk...\E[0m"
python -c "
try: import gtk; raise SystemExit(0)
except ImportError: raise SystemExit(-1)" 2&> /dev/null

if [ $? == 255 ]
then
  echo -e "\E[1m * Installing gtk...\E[0m"
  # Fetch, build, and install gtk.
  (  cd $CACHE
     curl
'https://pypi.python.org/packages/source/P/PyGTK/pygtk-
2.24.0.tar.bz2' > 'pygtk.tar.bz2'
     tar -xvf pygtk.tar.bz2
     (  cd pygtk*
        ./configure --prefix=$VIRTUAL_ENV
PKG_CONFIG_PATH=/usr/local/lib/pkgconfig:$VIRTUAL_ENV
/lib/pkgconfig
        make -j$CORES
        make install
     )
  )
Fi
```

Directory: intelora-core/setup/dev_setup.sh

```
set -Ee

if [ $(id -u) -eq 0 ]; then
  echo "This script should not be run as root or with sudo."
  exit 1
fi

TOP=$(cd $(dirname $0) && pwd -L)
VIRTUALENV_ROOT=${VIRTUALENV_ROOT:-
"${HOME}/.virtualenvs/mycroft"}

# create virtualenv, consistent with virtualenv-wrapper
conventions
if [ ! -d ${VIRTUALENV_ROOT} ]; then
  mkdir -p $(dirname ${VIRTUALENV_ROOT})
  virtualenv -p python2.7 ${VIRTUALENV_ROOT}
fi
source ${VIRTUALENV_ROOT}/bin/activate
cd ${TOP}
easy_install pip==7.1.2 # force version of pip
sudo pip install --upgrade virtualenv
```

```
# install requirements (except pocketsphinx)
pip2 install -r requirements.txt

CORES=$(nproc)
echo Building with $CORES cores.

TOP=$(cd $(dirname $0) && cd .. && pwd -L)
cd ${TOP}

# build and install pocketsphinx
#${TOP}/scripts/install-pocketsphinx.sh -q

# build and install mimic
${TOP}/scripts/install-mimic.sh

# install pygtk for desktop_launcher skill
${TOP}/scripts/install-pygtk.sh
```

Directory: intelora-core/setup/mycroft-base-MANIFEST.in

```
recursive-include mycroft/client/speech/model *
include requirements.txt
include mycroft/configuration/*.conf
recursive-include mycroft/skills/*/dialog *
recursive-include mycroft/skills/*/vocab *
recursive-include mycroft/skills/*/regex *
include mycroft/skills/alarm/alarm.mp3
include mycroft/skills/volume/blop-mark-diangelo.wav
#include mycroft/tts/mycroft_voice_4.0.flitevox
recursive-include mycroft/client/wifisetup/web/* *
include mycroft/client/wifisetup/web/index.html
```

Directory: intelora-core/setup/mycroft-base-setup.py

```
from setuptools import setup

from mycroft.util.setup_base import (
   find_all_packages,
   required,
   get_version,
   place_manifest
)

__author__ = 'seanfitz'

place_manifest('mycroft-base-MANIFEST.in')

setup(
   name="mycroft-core",
   version=get_version(),
   install_requires=[required('requirements.txt'), 'wifi'],
   packages=find_all_packages("mycroft"),
   include_package_data=True,

   entry_points={
      'console_scripts': [
         'mycroft-speech-
client=mycroft.client.speech.main:main',
         'mycroft-
messagebus=mycroft.messagebus.service.main:main',
         'mycroft-skills=mycroft.skills.main:main',
         'mycroft-echo-
observer=mycroft.messagebus.client.ws:echo',
         'mycroft-audio-test=mycroft.util.audio_test:main',
         'mycroft-enclosure-
client=mycroft.client.enclosure.main:main',
         'mycroft-wifi-setup-
client=mycroft.client.wifisetup.main:main',
         'mycroft-cli-client=mycroft.client.text.main:main'
      ]
   }
)
```

Directory: intelora-core/setup/requirement.txt

```
shortuuid==0.4.3
pystache==0.5.4
configobj==5.0.6
wikipedia==1.4.0
requests==2.8.1
pyOpenSSL==16.0.0
ndg-httpsclient==0.4.0
pyasn1==0.1.9
gTTS==1.0.7
backports.ssl-match-hostname==3.4.0.2
certifi==2016.2.28
PyAudio==0.2.8
pyee==1.0.1
SpeechRecognition==3.1.3
tornado==4.2.1
websocket-client==0.32.0
adapt-parser==0.2.7
pyowm==2.2.1
wolframalpha==1.4
futures==3.0.3
requests-futures==0.9.5
astral==0.9
tzlocal==1.2
parsedatetime==1.5
pdoc==0.3.2
pyyaml==3.11
feedparser==5.2.1
pyalsaaudio==0.8.2
xmlrunner==1.7.7
pyserial==3.0
netifaces==0.10.4
pyjokes==0.5.0
psutil==4.1.0
pep8==1.7.0
multi_key_dict==2.0.3
pocketsphinx==0.1.0
wifi==0.3.8
pyroute2==0.4.5
urllib5==5.0.0
pyric==0.1.6
inflection==0.3.1
```

Directory: intelora-core/setup/skills-sdk-MANIFEST.in

```
include requirements.txt
include mycroft/configuration/*.conf
include mycroft/util/setup_base.py
include mycroft/__version__.py
include skills-sdk-MANIFEST.in
```

Directory: intelora-core/setup/skills-sdk-setup.py

```
from setuptools import setup

from mycroft.util.setup_base import get_version,
place_manifest

__author__ = 'seanfitz'

place_manifest("skills-sdk-MANIFEST.in")

setup(
   name="mycroft-skills-sdk",
   version=get_version(),
   install_requires=[
      "mustache==0.1.4",
      "configobj==5.0.6",
      "pyee==1.0.1",
      "adapt-parser==0.2.1",
      "websocket-client==0.32.0"
   ],
   packages=[
      "mycroft.configuration",
      "mycroft.dialog",
      "mycroft.filesystem",
      "mycroft.messagebus",
      "mycroft.messagebus.client",
      "mycroft.session",
      "mycroft.skills.intent",
      "mycroft.skills",
      "mycroft.util",
      "mycroft"
   ],
   include_package_data=True,

   entry_points={
      'console_scripts': [
         'mycroft-skill-container=mycroft.skills.container:main'
      ]
   }
)
```

Directory: intelora-core/setup/test-requirement.txt

```
pep8
xmlrunner==1.7.7
```

Directory: intelora-core/test/mycroft.con

```
{
  "server": {
    "metrics": false
  }
}
```

Directory: intelora-core/test/main.py

```
import sys
from unittest import TestLoader

from os.path import dirname
from xmlrunner import XMLTestRunner

from mycroft.configuration import ConfigurationManager

__author__ = 'seanfitz, jdorleans'

if __name__ == "__main__":
   fail_on_error = "--fail-on-error" in sys.argv
   ConfigurationManager.load_local(['mycroft.conf'],
keep_user_config=False)

   tests = TestLoader().discover(dirname(__file__), "*.py")
   result = XMLTestRunner("./build/report/tests").run(tests)

   if fail_on_error and len(result.failures + result.errors) > 0:
      sys.exit(1)
```

Directory: intelora-core/test/test_runner.py

```
import sys
import unittest

from os.path import dirname
from xmlrunner import XMLTestRunner

from mycroft.configuration import ConfigurationManager

__author__ = 'seanfitz, jdorleans'
if __name__ == "__main__":
   fail_on_error = "--fail-on-error" in sys.argv
   ConfigurationManager.load_local(['mycroft.ini'])

   tests = unittest.TestLoader().discover(dirname(__file__),
"*.py")
   runner = XMLTestRunner("./build/report/tests")
   result = runner.run(tests)

   if fail_on_error and len(result.failures + result.errors) > 0:
      sys.exit(1)
```

Directory: intelora-core/test/aidio-accuracy-test

```
import os
import time
import wave
from glob import glob
from os.path import dirname, join

import pyee
from speech_recognition import AudioSource

from mycroft.client.speech.listener import RecognizerLoop
from mycroft.client.speech.mic import
ResponsiveRecognizer
from mycroft.client.speech.mic import logger as
speech_logger
from mycroft.util.log import getLogger

__author__ = 'wolfgange3311999'
logger = getLogger('audio_test_runner')


def to_percent(val):
   return "{0:.2f}".format(100.0 * val) + "%"
```

```
class FileStream(object):
    MIN_S_TO_DEBUG = 5.0

    # How long between printing debug info to screen
    UPDATE_INTERVAL_S = 1.0

    def __init__(self, file_name):
        self.file = wave.open(file_name, 'rb')
        self.size = self.file.getnframes()
        self.sample_rate = self.file.getframerate()
        self.sample_width = self.file.getsampwidth()
        self.last_update_time = 0.0

        self.total_s = self.size / self.sample_rate /
self.sample_width
        if self.total_s > self.MIN_S_TO_DEBUG:
            self.debug = True
        else:
            self.debug = False

    def calc_progress(self):
        return float(self.file.tell()) / self.size

    def read(self, chunk_size):

        progress = self.calc_progress()
        if progress == 1.0:
            raise EOFError

        if self.debug:
            cur_time = time.time()
            dt = cur_time - self.last_update_time
            if dt > self.UPDATE_INTERVAL_S:
                self.last_update_time = cur_time
                print(to_percent(progress))

        return self.file.readframes(chunk_size)

    def close(self):
        self.file.close()


class FileMockMicrophone(AudioSource):
    def __init__(self, file_name):
        self.stream = FileStream(file_name)
        self.SAMPLE_RATE = self.stream.sample_rate
        self.SAMPLE_WIDTH = self.stream.sample_width
        self.CHUNK = 1024

    def close(self):
        self.stream.close()


class AudioTester(object):
    def __init__(self, samp_rate):
        print  # Pad debug messages
        self.ww_recognizer =
RecognizerLoop.create_mycroft_recognizer(
            samp_rate, 'en-us')
        self.listener =
ResponsiveRecognizer(self.ww_recognizer)
        print
        speech_logger.setLevel(100)  # Disables logging to clean
output

    def test_audio(self, file_name):
        source = FileMockMicrophone(file_name)
        ee = pyee.EventEmitter()

        class SharedData:
            times_found = 0

        def on_found_wake_word():
            SharedData.times_found += 1

        ee.on('recognizer_loop:record_begin',
on_found_wake_word)

        try:
            while True:
                self.listener.listen(source, ee)
        except EOFError:
            pass

        return SharedData.times_found


class Color:
    BOLD = '\033[1m'
    NORMAL = '\033[0m'
    GREEN = '\033[92m'
    RED = '\033[91m'


def bold_str(val):
    return Color.BOLD + str(val) + Color.NORMAL


def get_root_dir():
    return dirname(dirname(__file__))


def get_file_names(folder):
    query = join(folder, '*.wav')
    root_dir = get_root_dir()
    full_path = join(root_dir, query)
    file_names = sorted(glob(full_path))

    if len(file_names) < 1:
        raise IOError

    return file_names


def test_audio_files(tester, file_names, on_file_finish):
    num_found = 0
    for file_name in file_names:
        short_name = os.path.basename(file_name)
        times_found = tester.test_audio(file_name)

        num_found += times_found
        on_file_finish(short_name, times_found)

    return num_found


def file_frame_rate(file_name):
    wf = wave.open(file_name, 'rb')
    frame_rate = wf.getframerate()
    wf.close()
```

```
    return frame_rate


def print_ww_found_status(word, short_name):
    print("Wake word " + bold_str(word) + " - " + short_name)


def test_false_negative(directory):
    file_names = get_file_names(directory)

    # Grab audio format info from first file
    tester = AudioTester(file_frame_rate(file_names[0]))

    def on_file_finish(short_name, times_found):
        not_found_str = Color.RED + "Not found"
        found_str = Color.GREEN + "Detected "
        status_str = not_found_str if times_found == 0 else
found_str
        print_ww_found_status(status_str, short_name)

    num_found = test_audio_files(tester, file_names,
on_file_finish)
    total = len(file_names)

    print
    print("Found " + bold_str(num_found) + " out of " +
bold_str(total))
    print(bold_str(to_percent(float(num_found) / total)) + "
accuracy.")
    print


def test_false_positive(directory):
    file_names = get_file_names(directory)

    # Grab audio format info from first file
    tester = AudioTester(file_frame_rate(file_names[0]))

    def on_file_finish(short_name, times_found):
        not_found_str = Color.GREEN + "Not found"
        found_str = Color.RED + "Detected "
        status_str = not_found_str if times_found == 0 else
found_str
        print_ww_found_status(status_str, short_name)

    num_found = test_audio_files(tester, file_names,
on_file_finish)
    total = len(file_names)

    print
    print("Found " + bold_str(num_found) + " false positives")
    print("in " + bold_str(str(total)) + " files")
    print


def run_test():
    directory = join('audio-accuracy-test', 'data')

    false_neg_dir = join(directory, 'with_wake_word',
'query_after')
    false_pos_dir = join(directory, 'without_wake_word')

    try:
        test_false_negative(false_neg_dir)
    except IOError:
        print(bold_str("Warning: No wav files found in " +
false_neg_dir))

    try:
        test_false_positive(false_pos_dir)
    except IOError:
        print(bold_str("Warning: No wav files found in " +
false_pos_dir))

    print("Complete!")


if __name__ == "__main__":
    run_test()
```

Directory: intelora-core/test/client/audio_consumer_test.py

```
import unittest
from Queue import Queue

import speech_recognition
from os.path import dirname, join
from speech_recognition import WavFile, AudioData

from mycroft.client.speech.listener import AudioConsumer,
RecognizerLoop
from mycroft.client.speech.local_recognizer import
LocalRecognizer
from mycroft.stt import GoogleSTT

__author__ = 'seanfitz'


class MockRecognizer(object):
    def __init__(self):
        self.transcriptions = []

    def recognize_google(self, audio, key=None,
language=None, show_all=False):
        if len(self.transcriptions) > 0:
            return self.transcriptions.pop(0)
        else:
            raise speech_recognition.UnknownValueError()

    def set_transcriptions(self, transcriptions):
        self.transcriptions = transcriptions


class AudioConsumerTest(unittest.TestCase):
    """
    AudioConsumerTest
    """

    def setUp(self):
        self.loop = RecognizerLoop()
        self.queue = Queue()
        self.recognizer = MockRecognizer()

        self.consumer = AudioConsumer(
            self.loop.state, self.queue, self.loop, GoogleSTT(),

LocalRecognizer(self.loop.wakeup_recognizer.key_phrase,
                            self.loop.wakeup_recognizer.phonemes,
"1e-16"),
```

```
        self.loop.mycroft_recognizer)

    def __create_sample_from_test_file(self, sample_name):
        root_dir = dirname(dirname(dirname(__file__)))
        filename = join(
            root_dir, 'test', 'client', 'data', sample_name + '.wav')
        wavfile = WavFile(filename)
        with wavfile as source:
            return AudioData(
                source.stream.read(), wavfile.SAMPLE_RATE,
                wavfile.SAMPLE_WIDTH)

    def test_word_extraction(self):
        """
        This is intended to test the extraction of the word:
``mycroft``.
        The values for ``ideal_begin`` and ``ideal_end`` were
found using an
        audio tool like Audacity and they represent a sample
value position of
        the audio. ``tolerance`` is an acceptable margin error for
the distance
        between the ideal and actual values found by the
``WordExtractor``
        """

        # TODO: implement WordExtractor test without relying
on the listener
        return

        audio =
self.__create_sample_from_test_file('weather_mycroft')
        self.queue.put(audio)
        tolerance = 4000
        ideal_begin = 70000
        ideal_end = 92000

        monitor = {}
        self.recognizer.set_transcriptions(["what's the weather
next week"])

        def wakeword_callback(message):
            monitor['pos_begin'] = message.get('pos_begin')
            monitor['pos_end'] = message.get('pos_end')

        self.loop.once('recognizer_loop:wakeword',
wakeword_callback)
        self.consumer.read()

        actual_begin = monitor.get('pos_begin')
        self.assertIsNotNone(actual_begin)
        diff = abs(actual_begin - ideal_begin)
        self.assertTrue(
            diff <= tolerance,
            str(diff) + " is not less than " + str(tolerance))

        actual_end = monitor.get('pos_end')
        self.assertIsNotNone(actual_end)
        diff = abs(actual_end - ideal_end)
        self.assertTrue(
            diff <= tolerance,
            str(diff) + " is not less than " + str(tolerance))

    def test_wakeword_in_beginning(self):

self.queue.put(self.__create_sample_from_test_file('weathe
r_mycroft'))
        self.recognizer.set_transcriptions(["what's the weather
next week"])
        monitor = {}

        def callback(message):
            monitor['utterances'] = message.get('utterances')

        self.loop.once('recognizer_loop:utterance', callback)
        self.consumer.read()

        utterances = monitor.get('utterances')
        self.assertIsNotNone(utterances)
        self.assertTrue(len(utterances) == 1)
        self.assertEquals("what's the weather next week",
utterances[0])

    def test_wakeword(self):

self.queue.put(self.__create_sample_from_test_file('mycrof
t'))
        self.recognizer.set_transcriptions(["silence"])
        monitor = {}

        def callback(message):
            monitor['utterances'] = message.get('utterances')

        self.loop.once('recognizer_loop:utterance', callback)
        self.consumer.read()

        utterances = monitor.get('utterances')
        self.assertIsNotNone(utterances)
        self.assertTrue(len(utterances) == 1)
        self.assertEquals("silence", utterances[0])

    def test_ignore_wakeword_when_sleeping(self):

self.queue.put(self.__create_sample_from_test_file('mycrof
t'))
        self.recognizer.set_transcriptions(["not detected"])
        self.loop.sleep()
        monitor = {}

        def wakeword_callback(message):
            monitor['wakeword'] = message.get('utterance')

        self.loop.once('recognizer_loop:wakeword',
wakeword_callback)
        self.consumer.read()
        self.assertIsNone(monitor.get('wakeword'))
        self.assertTrue(self.loop.state.sleeping)

    def test_wakeup(self):

self.queue.put(self.__create_sample_from_test_file('mycrof
t_wakeup'))
        self.loop.sleep()
        self.consumer.read()
        self.assertFalse(self.loop.state.sleeping)

    def test_stop(self):

self.queue.put(self.__create_sample_from_test_file('mycrof
t'))
```

```
        self.consumer.read()


self.queue.put(self.__create_sample_from_test_file('stop'))
        self.recognizer.set_transcriptions(["stop"])
        monitor = {}

        def utterance_callback(message):
            monitor['utterances'] = message.get('utterances')

        self.loop.once('recognizer_loop:utterance',
utterance_callback)
        self.consumer.read()

        utterances = monitor.get('utterances')
        self.assertIsNotNone(utterances)
        self.assertTrue(len(utterances) == 1)
        self.assertEquals("stop", utterances[0])

    def test_record(self):

self.queue.put(self.__create_sample_from_test_file('mycrof
t'))
        self.consumer.read()


self.queue.put(self.__create_sample_from_test_file('record'
))
        self.recognizer.set_transcriptions(["record"])
        monitor = {}

        def utterance_callback(message):
            monitor['utterances'] = message.get('utterances')

        self.loop.once('recognizer_loop:utterance',
utterance_callback)
        self.consumer.read()

        utterances = monitor.get('utterances')
        self.assertIsNotNone(utterances)
        self.assertTrue(len(utterances) == 1)
        self.assertEquals("record", utterances[0])
```

Directory: intelora-core/test/client/dynamic_energy_test.py

```
import unittest
import audioop
from speech_recognition import AudioSource
from mycroft.client.speech.mic import
ResponsiveRecognizer

__author__ = 'seanfitz'


class MockStream(object):
    def __init__(self):
        self.chunks = []

    def inject(self, chunk):
        self.chunks.append(chunk)

    def read(self, chunk_size):
        result = self.chunks[0]
        if len(self.chunks) > 1:
            self.chunks = self.chunks[1:]
        return result


class MockSource(AudioSource):
    def __enter__(self):
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        pass

    def __init__(self, stream=None):
        self.stream = stream if stream else MockStream()
        self.CHUNK = 1024
        self.SAMPLE_RATE = 16000
        self.SAMPLE_WIDTH = 2


class DynamicEnergytest(unittest.TestCase):
    def setUp(self):
        pass

    def testMaxAudioWithBaselineShift(self):
        low_base = b"".join(["\x10\x00\x01\x00"] * 100)
        higher_base = b"".join(["\x01\x00\x00\x01"] * 100)

        source = MockSource()

        for i in xrange(100):
            source.stream.inject(low_base)

        source.stream.inject(higher_base)
        recognizer = ResponsiveRecognizer(None)

        sec_per_buffer = float(source.CHUNK) /
(source.SAMPLE_RATE *
                               source.SAMPLE_WIDTH)

        test_seconds = 30.0
        while test_seconds > 0:
            test_seconds -= sec_per_buffer
            data = source.stream.read(source.CHUNK)
            energy = recognizer.calc_energy(data,
source.SAMPLE_WIDTH)
            recognizer.adjust_threshold(energy, sec_per_buffer)

        higher_base_energy = audioop.rms(higher_base,
source.SAMPLE_WIDTH)
        # after recalibration (because of max audio length) new
threshold
        # should be >= 1.5 * higher_base_energy
        delta_below_threshold = (
            recognizer.energy_threshold - higher_base_energy)
        min_delta = higher_base_energy * .5
        assert abs(delta_below_threshold - min_delta) < 1
```

Directory: intelora-core/test/client/local_recognizer_test.py

```
import unittest

import os
from speech_recognition import WavFile

from mycroft.client.speech.listener import RecognizerLoop
```

```
__author__ = 'seanfitz'

DATA_DIR =
os.path.join(os.path.abspath(os.path.dirname(__file__)),
"data")


class LocalRecognizerTest(unittest.TestCase):
    def setUp(self):
        self.recognizer =
RecognizerLoop.create_mycroft_recognizer(16000,
                                                 "en-us")

    def testRecognizerWrapper(self):
        source = WavFile(os.path.join(DATA_DIR,
"hey_mycroft.wav"))
        with source as audio:
            hyp = self.recognizer.transcribe(audio.stream.read())
            assert self.recognizer.key_phrase in
hyp.hypstr.lower()
        source = WavFile(os.path.join(DATA_DIR,
"mycroft.wav"))
        with source as audio:
            hyp = self.recognizer.transcribe(audio.stream.read())
            assert self.recognizer.key_phrase in
hyp.hypstr.lower()

    def testRecognitionInLongerUtterance(self):
        source = WavFile(os.path.join(DATA_DIR,
"weather_mycroft.wav"))
        with source as audio:
            hyp = self.recognizer.transcribe(audio.stream.read())
            assert self.recognizer.key_phrase in
hyp.hypstr.lower()
```

Directory: intelora-core/test/configuration/__init__.py

```
import unittest

from os.path import dirname, join

from mycroft.configuration import ConfigurationLoader,
ConfigurationManager, \
    DEFAULT_CONFIG, SYSTEM_CONFIG, USER_CONFIG,
RemoteConfiguration

__author__ = 'jdorleans'


class AbstractConfigurationTest(unittest.TestCase):
    def setUp(self):
        self.config_path = join(dirname(__file__),
'mycroft.conf')

    @staticmethod
    def create_config(lang='en-us', module='mimic',
voice="ap"):
        config = {
            'lang': lang,
            'tts': {
                'module': module,
                module: {'voice': voice}
            }
        }
        return config

    def assert_config(self, config, lang='en-us',
module='mimic', voice="ap"):
        self.assertIsNotNone(config)
        lan = config.get('lang')
        self.assertIsNotNone(lan)
        self.assertEquals(lan, lang)
        tts = config.get('tts')
        self.assertIsNotNone(tts)
        mod = tts.get('module')
        self.assertEquals(mod, module)
        voi = tts.get(mod, {}).get("voice")
        self.assertEquals(voi, voice)


class ConfigurationLoaderTest(AbstractConfigurationTest):
    def test_init_config_with_defaults(self):
        self.assertEquals(ConfigurationLoader.init_config(), {})

    def test_init_config_with_new_config(self):
        config = {'a': 'b'}

self.assertEquals(ConfigurationLoader.init_config(config),
config)

    def test_init_locations_with_defaults(self):
        locations = [DEFAULT_CONFIG, SYSTEM_CONFIG,
USER_CONFIG]
        self.assertEquals(ConfigurationLoader.init_locations(),
locations)

    def test_init_locations_with_new_location(self):
        locations = [self.config_path]

self.assertEquals(ConfigurationLoader.init_locations(location
s),
                          locations)

    def test_validate_data(self):
        try:
            ConfigurationLoader.validate({}, [])
        except TypeError:
            self.fail()

    def test_validate_data_with_invalid_data(self):
        self.assertRaises(TypeError,
ConfigurationLoader.validate)

    def test_load(self):
        self.assert_config(ConfigurationLoader.load())

    def test_load_with_override_custom(self):
        config = self.create_config('pt-br', 'espeak', 'f1')
        config = ConfigurationLoader.load(config)
        self.assert_config(config)

    def test_load_with_override_default(self):
        config = self.create_config()
        config = ConfigurationLoader.load(config,
[self.config_path])
        self.assert_config(config, 'pt-br', 'espeak', 'f1')

    def test_load_with_extra_custom(self):
        my_config = {'key': 'value'}
```

```
        config = ConfigurationLoader.load(my_config)
        self.assert_config(config)

        value = config.get('key', None)
        self.assertIsNotNone(value)
        self.assertEquals(value, my_config.get('key'))

    def test_load_with_invalid_config_type(self):
        self.assertRaises(TypeError, ConfigurationLoader.load,
'invalid_type')

    def test_load_with_invalid_locations_type(self):
        self.assertRaises(TypeError, ConfigurationLoader.load,
                          None, self.config_path)

    def test_load_with_invalid_locations_path(self):
        locations = ['./invalid/mycroft.conf',
'./invalid_mycroft.conf']
        config = ConfigurationLoader.load(None, locations,
False)
        self.assertEquals(config, {})


class RemoteConfigurationTest(AbstractConfigurationTest):
    def test_validate_config(self):
        try:
            RemoteConfiguration.validate(self.create_config())
        except TypeError:
            self.fail()

    def test_validate_config_with_invalid_config(self):
        self.assertRaises(TypeError,
RemoteConfiguration.validate)

    def test_load_without_remote_config(self):
        config = self.create_config()
        self.assertEquals(RemoteConfiguration.load(config),
config)


class ConfigurationManagerTest(AbstractConfigurationTest):
    def test_load_defaults(self):
        ConfigurationManager.load_defaults()

self.assert_config(ConfigurationManager.load_defaults())

    def test_load_local(self):
        ConfigurationManager.load_defaults()
        self.assert_config(ConfigurationManager.load_local())

    def test_load_local_with_locations(self):
        ConfigurationManager.load_defaults()
        config =
ConfigurationManager.load_local([self.config_path])
        self.assert_config(config, 'pt-br', 'espeak', 'f1')

    def test_load_remote(self):
        ConfigurationManager.load_defaults()

self.assert_config(ConfigurationManager.load_remote())

    def test_get(self):
        ConfigurationManager.load_defaults()
        self.assert_config(ConfigurationManager.get())

    def test_load_get_with_locations(self):
        ConfigurationManager.load_defaults()
        config = ConfigurationManager.get([self.config_path])
        self.assert_config(config, 'pt-br', 'espeak', 'f1')
```

Directory: intelora-core/test/configuration/mycroft.conf

```
{
  "lang": "pt-br",
  "tts": {
    "module": "espeak",
    "espeak": {
      "voice": "f1"
    }
  }
}
```

Directory: intelora-core/test/regex_test/valid/multiple.rx

```
(?P<MultipleTest1>.*)
(?P<MultipleTest2>.*)
```

Directory: intelora-core/test/regex_test/valid/single.rx

```
(?P<SingleTest>.*)
```

Directory: intelora-core/test/skills/__init__.py

```
__author__ = 'seanfitz'
```

Directory: intelora-core/test/skills/core.py

```
import unittest

from os.path import join, dirname, abspath
from re import error

from mycroft.skills.core import load_regex_from_file,
load_regex, \
    load_vocab_from_file, load_vocabulary
from mycroft.util.log import getLogger

__author__ = 'eward'
logger = getLogger(__name__)


class MockEmitter(object):
    def __init__(self):
        self.reset()

    def emit(self, message):
        self.types.append(message.type)
        self.results.append(message.data)

    def get_types(self):
        return self.types

    def get_results(self):
        return self.results

    def reset(self):
        self.types = []
```

```
        self.results = []


class MycroftSkillTest(unittest.TestCase):
    emitter = MockEmitter()
    regex_path = abspath(join(dirname(__file__),
'../regex_test'))
    vocab_path = abspath(join(dirname(__file__),
'../vocab_test'))

    def check_vocab_from_file(self, filename,
vocab_type=None, result_list=[]):
        load_vocab_from_file(join(self.vocab_path, filename),
vocab_type,
                             self.emitter)
        self.check_emitter(result_list)

    def check_regex_from_file(self, filename, result_list=[]):
        load_regex_from_file(join(self.regex_path, filename),
self.emitter)
        self.check_emitter(result_list)

    def check_vocab(self, path, result_list=[]):
        load_vocabulary(path, self.emitter)
        self.check_emitter(result_list)

    def check_regex(self, path, result_list=[]):
        load_regex(path, self.emitter)
        self.check_emitter(result_list)

    def check_emitter(self, result_list):
        for type in self.emitter.get_types():
            self.assertEquals(type, 'register_vocab')
        self.assertEquals(sorted(self.emitter.get_results()),
                          sorted(result_list))
        self.emitter.reset()

    def test_load_regex_from_file_single(self):
        self.check_regex_from_file('valid/single.rx',
                                   [{'regex': '(?P<SingleTest>.*)'}])

    def test_load_regex_from_file_multiple(self):
        self.check_regex_from_file('valid/multiple.rx',
                                   [{'regex': '(?P<MultipleTest1>.*)'},
                                    {'regex': '(?P<MultipleTest2>.*)'}])

    def test_load_regex_from_file_none(self):
        self.check_regex_from_file('invalid/none.rx')

    def test_load_regex_from_file_invalid(self):
        try:
            self.check_regex_from_file('invalid/invalid.rx')
        except error as e:
            self.assertEquals(e.__str__(),
                              'unexpected end of regular expression')

    def test_load_regex_from_file_does_not_exist(self):
        try:
            self.check_regex_from_file('does_not_exist.rx')
        except IOError as e:
            self.assertEquals(e.strerror, 'No such file or directory')

    def test_load_regex_full(self):
        self.check_regex(join(self.regex_path, 'valid'),
                         [{'regex': '(?P<MultipleTest1>.*)'},
                          {'regex': '(?P<MultipleTest2>.*)'},
                          {'regex': '(?P<SingleTest>.*)'}])

    def test_load_regex_empty(self):
        self.check_regex(join(dirname(__file__),
                              'wolfram_alpha'))

    def test_load_regex_fail(self):
        try:
            self.check_regex(join(dirname(__file__),
                                  'regex_test_fail'))
        except OSError as e:
            self.assertEquals(e.strerror, 'No such file or directory')

    def test_load_vocab_from_file_single(self):
        self.check_vocab_from_file('valid/single.voc',
'test_type',
                                   [{'start': 'test', 'end': 'test_type'}])

    def test_load_vocab_from_file_single_alias(self):
        self.check_vocab_from_file('valid/singlealias.voc',
'test_type',
                                   [{'start': 'water', 'end': 'test_type'},
                                    {'start': 'watering', 'end': 'test_type',
                                     'alias_of': 'water'}])

    def test_load_vocab_from_file_multiple(self):
        self.check_vocab_from_file('valid/multiple.voc',
'test_type',
                                   [{'start': 'animal', 'end': 'test_type'},
                                    {'start': 'animals', 'end': 'test_type'}])

    def test_load_vocab_from_file_multiple_alias(self):
        self.check_vocab_from_file('valid/multiplealias.voc',
'test_type',
                                   [{'start': 'chair', 'end': 'test_type'},
                                    {'start': 'chairs', 'end': 'test_type',
                                     'alias_of': 'chair'},
                                    {'start': 'table', 'end': 'test_type'},
                                    {'start': 'tables', 'end': 'test_type',
                                     'alias_of': 'table'}])

    def test_load_vocab_from_file_none(self):
        self.check_vocab_from_file('none.voc')

    def test_load_vocab_from_file_does_not_exist(self):
        try:
            self.check_vocab_from_file('does_not_exist.voc')
        except IOError as e:
            self.assertEquals(e.strerror, 'No such file or directory')

    def test_load_vocab_full(self):
        self.check_vocab(join(self.vocab_path, 'valid'),
                         [{'start': 'test', 'end': 'single'},
                          {'start': 'water', 'end': 'singlealias'},
                          {'start': 'watering', 'end': 'singlealias',
                           'alias_of': 'water'},
                          {'start': 'animal', 'end': 'multiple'},
                          {'start': 'animals', 'end': 'multiple'},
                          {'start': 'chair', 'end': 'multiplealias'},
                          {'start': 'chairs', 'end': 'multiplealias',
                           'alias_of': 'chair'},
                          {'start': 'table', 'end': 'multiplealias'},
                          {'start': 'tables', 'end': 'multiplealias',
                           'alias_of': 'table'}])

    def test_load_vocab_empty(self):
```

```
    self.check_vocab(join(dirname(__file__),
'wolfram_alpha'))

  def test_load_vocab_fail(self):
    try:
      self.check_regex(join(dirname(__file__),
                    'vocab_test_fail'))
    except OSError as e:
      self.assertEquals(e.strerror, 'No such file or directory')
```

Directory: intelora-core/test/skills/discover_tests.py

```
import os
import glob
import unittest
from test.skills.skill_tester import MockSkillsLoader, SkillTest

__author__ = 'seanfitz'

PROJECT_ROOT =
os.path.dirname(os.path.dirname(os.path.dirname(__file__)
))


def discover_tests():
  tests = {}
  skills = [
    skill for skill
    in glob.glob(os.path.join(PROJECT_ROOT,
'mycroft/skills/*'))
    if os.path.isdir(skill)
  ]

  for skill in skills:
    test_intent_files = [
      f for f
      in glob.glob(os.path.join(skill,
'test/intent/*.intent.json'))
    ]
    if len(test_intent_files) > 0:
      tests[skill] = test_intent_files

  return tests


class IntentTestSequenceMeta(type):
  def __new__(mcs, name, bases, d):
    def gen_test(a, b):
      def test(self):
        SkillTest(a, b, self.emitter).run()
      return test

    tests = discover_tests()
    for skill in tests.keys():
      skill_name = os.path.basename(skill)
      for example in tests[skill]:
        example_name = os.path.basename(
          os.path.splitext(os.path.splitext(example)[0])[0])
        test_name = "test_IntentValidation[%s:%s]" %
(skill_name,
                                example_name)
        d[test_name] = gen_test(skill, example)
    return type.__new__(mcs, name, bases, d)


class IntentTestSequence(unittest.TestCase):
  __metaclass__ = IntentTestSequenceMeta

  def setUp(self):
    self.emitter = MockSkillsLoader(
      os.path.join(PROJECT_ROOT, 'mycroft',
'skills')).load_skills()

if __name__ == '__main__':
  unittest.main()
```

Directory: intelora-core/test/skills/scheduled_skills.py

```
from datetime import datetime, timedelta
import unittest

from mycroft.skills.scheduled_skills import ScheduledSkill
from mycroft.util.log import getLogger

__author__ = 'eward'

logger = getLogger(__name__)


class ScheduledSkillTest(unittest.TestCase):
  skill = ScheduledSkill(name='ScheduledSkillTest')

  def test_formatted_time_today_hours(self):
    date = datetime.now() + timedelta(hours=2)
    self.assertEquals(self.skill.

get_formatted_time(float(date.strftime('%s'))),
              "1 hours and 59 minutes from now")

  def test_formatted_time_today_min(self):
    date = datetime.now() + timedelta(minutes=2)
    self.assertEquals(self.skill.

get_formatted_time(float(date.strftime('%s'))),
              "1 minutes and 59 seconds from now")

  def test_formatted_time_days(self):
    date = datetime.now() + timedelta(days=2)
    self.assertEquals(self.skill.

get_formatted_time(float(date.strftime('%s'))),
              date.strftime("%A, %B %d, %Y at %H:%M"))
```

Directory: intelora-core/test/skills/skills_tester.py

```
import json

from pyee import EventEmitter

from mycroft.messagebus.message import Message
from mycroft.skills.core import load_skills

__author__ = 'seanfitz'


class RegistrationOnlyEmitter(object):
  def __init__(self):
```

```
        self.emitter = EventEmitter()

    def on(self, event, f):
        if event in [
            'register_intent',
            'register_vocab',
            'recognizer_loop:utterance'
        ]:
            self.emitter.on(event, f)

    def emit(self, event, *args, **kwargs):
        event_name = event.type
        self.emitter.emit(event_name, event, *args, **kwargs)


class MockSkillsLoader(object):
    def __init__(self, skills_root):
        self.skills_root = skills_root
        self.emitter = RegistrationOnlyEmitter()

    def load_skills(self):
        load_skills(self.emitter, self.skills_root)
        return self.emitter.emitter  # kick out the underlying
emitter


class SkillTest(object):
    def __init__(self, skill, example, emitter):
        self.skill = skill
        self.example = example
        self.emitter = emitter
        self.returned_intent = False

    def compare_intents(self, expected, actual):
        for key in expected.keys():
            if actual.get(key, "").lower() != expected.get(key,
"").lower():
                print(
                    "Expected %s: %s, Actual: %s" % (key,
expected.get(key),
                                                    actual.get(key)))
                assert False

    def run(self):
        example_json = json.load(open(self.example, 'r'))
        event = {'utterances': [example_json.get('utterance')]}

        def compare(intent):
            self.compare_intents(example_json.get('intent'),
intent.data)
            self.returned_intent = True

        self.emitter.once(example_json.get('intent_type'),
compare)
        self.emitter.emit(
            'recognizer_loop:utterance',
            Message('recognizer_loop:utterance', event))
        if not self.returned_intent:
            print("No intent handled")
            assert False
```

Directory: intelora-core/test/skills/pairing/__init__.py

```
import unittest

from mycroft.skills.pairing import PairingSkill


class PairingSkillTest(unittest.TestCase):
    skill = PairingSkill()
```

Directory: intelora-core/test/skills/wolfram_alpha/__init__.py

```
import unittest
import wolframalpha
from StringIO import StringIO

from mycroft.skills.wolfram_alpha import WolframAlphaSkill
from mycroft.util.log import getLogger

__author__ = 'eward'

logger = getLogger(__name__)


class WolframAlphaTest(unittest.TestCase):
    skill = WolframAlphaSkill()

    @staticmethod
    def format_result(pod_id, text, pod_num):
        return "<queryresult>\
        <pod id='" + pod_id + "' title = '" + pod_id + \
            "' position='" + pod_num + "'><subpod> \
        <plaintext>" + text +
"</plaintext></subpod></pod></queryresult>"

    @staticmethod
    def format_did_you_mean(text):
        tree = "<queryresult><didyoumeans>"
        for result in text:
            tree += "<didyoumean>" + result + "</didyoumean>"
        tree += "</didyoumeans></queryresult>"
        return tree

    def create_result(self, pod_id, value, pod_num):
        result = self.format_result(pod_id, value, pod_num)
        return wolframalpha.Result(StringIO(result))

    def create_did_you_mean(self, text):
        test = self.format_did_you_mean(text)
        return wolframalpha.Result(StringIO(test))

    def test_result_pod(self):
        res = self.create_result("Result", "7", '300')
        self.assertEquals(self.skill.get_result(res), "7")

    def test_value_pod(self):
        res = self.create_result("Value", "2^3", '300')
        self.assertEquals(self.skill.get_result(res), "2^3")

    def test_notable_facts_pod(self):
        res = self.create_result("NotableFacts:PeopleData",
                                 "PeopleData", '300')
        self.assertEquals(self.skill.get_result(res), "PeopleData")

    def test_basic_information_pod(self):
        res = self.create_result("BasicInformation:PeopleData",
                                 "Born in 1997", '300')
```

```
        self.assertEquals(self.skill.get_result(res), "Born in
1997")

    def test_decimal_approximation_pod(self):
        res = self.create_result("DecimalApproximation",
"5.6666666666", '300')
        self.assertEquals(self.skill.get_result(res), "5.666")

    def test_definition_pod(self):
        res = self.create_result("Definition:WordData",
                                 "a cat is a feline", '300')
        self.assertEquals(self.skill.get_result(res),
                          "a cat is a feline")

    def test_numbered_pod(self):
        res = self.create_result("MathConcept", "tangrams are
objects", '200')
        self.assertEqual(self.skill.get_result(res),
                         "tangrams are objects")

    def test_invalid_pod(self):
        res = self.create_result("InvalidTitle", "Test", '300')
        self.assertEquals(self.skill.get_result(res), None)

    def test_whitespace_process(self):
        self.assertEquals(self.skill.process_wolfram_string
                          ("Test     string"), "Test string")

    def test_pipe_process(self):
        self.assertEquals(self.skill.process_wolfram_string
                          ("Test | string"), "Test, string")

    def test_newline_process(self):
        self.assertEquals(self.skill.process_wolfram_string
                          ("Test\nstring"), "Test, string")

    def test_factorial_process(self):
        self.assertEquals(self.skill.process_wolfram_string
                          ("Test!"), "Test,factorial")

    def test_find_did_you_mean_exists(self):
        values = ['search for power', 'power']
        res = self.create_did_you_mean(values)
        self.assertEquals(self.skill._find_did_you_mean(res),
                          values)

    def test_find_did_you_mean_none(self):
        res = self.create_did_you_mean([])
        self.assertEquals(self.skill._find_did_you_mean(res), [])
```

Directory: intelora-core/test/skills/wolfram_alpha/english_question_parser_test.py

```
import unittest

from mycroft.skills.wolfram_alpha import
EnglishQuestionParser

__author__ = 'wolfgange3311999'


class EnglishQuestionParserTest(unittest.TestCase):
    parser = EnglishQuestionParser()
    test_jsons = [
        {
            'utterance': 'who is abraham lincoln',
            'parsed_question': {
                'QuestionWord': 'who',
                'QuestionVerb': 'is',
                'Query': 'abraham lincoln'
            }
        },
        {
            'utterance': 'what are they',
            'parsed_question': {
                'QuestionWord': 'what',
                'QuestionVerb': 'are',
                'Query': 'they'
            }
        },
        {
            'utterance': 'when was this',
            'parsed_question': {
                'QuestionWord': 'when',
                'QuestionVerb': 'was',
                'Query': 'this'
            }
        },
        {
            'utterance': 'why were they there',
            'parsed_question': {
                'QuestionWord': 'why',
                'QuestionVerb': 'were',
                'Query': 'they there'
            }
        },
        {
            'utterance': 'which is the thing',
            'parsed_question': {
                'QuestionWord': 'which',
                'QuestionVerb': 'is',
                'Query': 'the thing'
            }
        },
        {
            'utterance': 'who saw abraham lincoln',
            'parsed_question': {
                'QuestionWord': 'who',
                'QuestionVerb': 'saw',
                'Query': 'abraham lincoln'
            }
        },
        {
            'utterance': 'what began life',
            'parsed_question': {
                'QuestionWord': 'what',
                'QuestionVerb': 'began',
                'Query': 'life'
            }
        },
        {
            'utterance': 'where sat the person',
            'parsed_question': {
                'QuestionWord': 'where',
                'QuestionVerb': 'sat',
                'Query': 'the person'
            }
        },
        {
            'utterance': 'i like stuff',
```

```
            'parsed_question': None
        },
        {
            'utterance': 'what\'s a dog',
            'parsed_question': {
                'QuestionWord': 'what',
                'QuestionVerb': '\'s',
                'Query': 'a dog'
            }
        },
        {
            'utterance': 'who did this',
            'parsed_question': {
                'QuestionWord': 'who',
                'QuestionVerb': 'did',
                'Query': 'this'
            }
        }
    ]

    def test_question_parsing(self):
        for test_json in self.test_jsons:
            parsed_question =
self.parser.parse(test_json['utterance'])
            self.assertEquals(parsed_question,
test_json['parsed_question'])
```