```bash
#!/bin/bash

# This UNIX shell script makes graphical Vim use one instance where possible.
# Save this script as  /bin/gvim, make sure it's executable and you're off!

# Configuration variables.
GVIM=/usr/local/bin/gvim
WMCTRL=/usr/bin/wmctrl # <- try 'sudo apt-get install wmctrl'
DEBUG=no # yes or no

# Look for command-line arguments that aren't filenames and
# if one is found then fall back to Vim's regular behavior.
for ARG in "$@"; do
  if echo "$ARG" | grep -q '^note:'; then
    # Don't consider "note:..." as non-file argument (this is a
    # hack to make my note taking plug-in work with this script).
    continue
  elif [ ! -f "$ARG" ]; then
    # Replace the wrapper with Vim itself.
    [ $DEBUG = yes ] && echo "Process $$: Executing $GVIM $@" >&2
    exec $GVIM "$@"
    exit # <- shouldn't be reached!
  fi
done

# Check if Vim is already running.
if ! pidof $GVIM > /dev/null; then
  # It's not: Open file(s) in new instance.
  [ $DEBUG = yes ] && "Process $$: Executing $GVIM -p $@" >&2
  exec "$GVIM" -p "$@"
else
  [ $DEBUG = yes ] && echo "Process $$: Requesting server list" >&2
  $GVIM --serverlist | grep '^GVIM[0-9]*$' | while read VIM_SERVER_NAME; do
    [ $DEBUG = yes ] && echo "Process $$: Raising selected Vim server" >&2
    $WMCTRL -xa $VIM_SERVER_NAME
    # Pass any pathname arguments on to Vim?
    if [ $# -ge 1 ]; then
      # Open files on command-line in existing Vim window.
      [ $DEBUG = yes ] && echo "Process $$: Executing $GVIM --servername '$VIM_SERVER_NAME' --remote-tab-si
      exec $GVIM --servername "$VIM_SERVER_NAME" --remote-tab-silent "$@"
    fi
  done
fi
```