WebExp: A Java Toolbox for Web-Based Psychological Experiments*

Users' Guide for WebExp 2.1

Frank Keller
Human Communication Research Centre
University of Edinburgh
`keller@cogsci.ed.ac.uk`

Martin Corley
Department of Psychology
University of Edinburgh
`martin.corley@ed.ac.uk`

Steffan Corley
Sharp Laboratories of Europe
Oxford
`steffan@sharp.co.uk`

Lars Konieczny
Department of Computational Linguistics
University of the Saarland
`lkon@coli.uni-sb.de`

Amalia Todirascu
Department of Computer Science
Al. I. Cuza University of Iasi
`amaliat@iit.tuiasi.ro`

November 26, 1998

*Each of the authors of this documentation was involved in the design and implementation of WebExp. The documentation was written by Frank Keller, based on previous documents by Steffan Corley and Amalia Todirascu. All other authors are given in alphabetical order. The initial idea of conducting psychological experiments over the Web is due to Steffan Corley, who also implemented the first version of WebExp.

Technical Report HCRC/TR-99, Human Communication Research Centre,
University of Edinburgh, 1998.

Abstract

This User's Guide explains the installation and use of WebExp, a set of Java classes for conducting psychological experiments over the Word Wide Web.

The WebExp toolbox consists of two modules: the WebExp server, which is a stand-alone Java application, and the WebExp client, which is implemented as a Java applet. The server application runs on the Web server that hosts the experiment, and waits for client applets to connect to it. The client runs remotely on the machine of the experimental participant. It administers the experiment and connects to the server application to download the experimental stimuli, and to store the subject's responses.

WebExp offers the following features for conducting Web-based experiments:

- Two experimental paradigms are supported: magnitude estimation and sentence completion. Both within-subject and between-subject designs can be used.
- Automatic subject authentication is achieved by conducting basic plausibility checks on the subject's data and by verifying the subject's e-mail address.
- WebExp automatically creates an individual randomization of the experimental materials for each subject. The experimenter can impose constraints on the randomization to prevent certain experimental items from occurring consecutively.
- WebExp records the time a subject takes to respond to each experimental item. Automatic checks can be carried out on both onset times and completion times.
- LaTeX files can be generated from the experimental materials to be used as off-line questionnaires.
- The response data are stored in a format that can be easily processed by standard statistics packages.

Contents

1 General Overview **4**
1.1 Features 4
1.1.1 Implemented Features 4
1.1.2 Desirable Features 5
1.1.3 Known Problems 6
1.2 Web-Based Experiments 6
1.2.1 Pros of Web-Based Experiments 6
1.2.2 Cons of Web-Based Experiments 6

2 Using and Customizing the Toolbox **7**
2.1 Components of WebExp 7
2.2 Installation 7
2.3 Designing an Experiment 8
2.3.1 Experimental Paradigms 8
2.3.2 Randomization 8
2.3.3 Stimulus Files 9
2.3.4 Web Pages 10
2.3.5 Other Files 10
2.3.6 Applet Parameters 11
2.3.7 Randomization Parameters 14
2.4 Running an Experiment 15
2.4.1 Server Application 15
2.4.2 Client Applet 16
2.5 Response Files 16

3 Copyright Notice **18**

Chapter 1

General Overview

1.1 Features

This section reviews the main functionality of WebExp, a Java toolbox for Web-based psychological experiments.

1.1.1 Implemented Features

Here is a list of the features currently supported by WebExp:

- **Experimental Paradigms.** Two types of experimental paradigms are implemented:
  - **Magnitude Estimation.** This paradigm can be used for experiments involving numeric judgments, for example of linguistic stimuli. After having judged a modulus item, the subject has to provide a judgment for each experimental item relative to the modulus. A possible application for this is magnitude estimation of linguistic acceptability.
  - **Sentence Completion.** In this paradigm, the subject is presented with an experimental item, and has to provide a textual answer. This paradigm can be used for sentence completion studies, for instance.
- **Subject Authentication.** The experiment is designed to be completed by remote subjects—the idea is that the experimenter just sends out a message asking for subjects, and includes the URL of the experiment in the message. In this case, it becomes important to verify that subjects only attempt the experiment once, and that they fulfil all criteria. This is partially achieved by asking for the subject's e-mail address. WebExp conducts basic validity checks on this address, and also ensures that each address is used only once. Furthermore, the subject is sent a "thank you" e-mail at the end of the experiment, apparently from the experimenter. If the subject enters an invalid e-mail address, this message bounces and is received by the experimenter, who can then remove the subject's data.

  Of course, the experimental software can also be used under laboratory conditions where such subject authentication is not necessary.
- **Multiple Variants of an Experiment.** WebExp allows the specification of a number of different variants of a given experiment, and it guarantees that each variant will be

delivered to an equal number of participants. This feature is useful if the experiment uses a between-subject design, an application being different lexicalizations for the materials in a psycholinguistic experiment.

- **Randomization.** WebExp presents the items in the stimulus file (or files, if there are several variants) in an individual randomized order for each subject taking the experiment. The randomizer generates these orderings by randomly interleaving the items of each block in the stimulus file. Different constraints on randomization can be imposed. In the present implementation, the following types of randomizations are supported:
  - **Free Randomization.** The experimental items can appear in any order, blocking doesn't play a role.
  - **Constrained Randomization.** The experimenter specifies restrictions on how many items from which block (or category) can be displayed consecutively.
  - **Blocked Randomization.** The order of blocks is randomized, while the items in each block are presented in a fixed order.
- **Response Timing.** WebExp records the time it took for the subject to make a first key press (the onset time) and to type the complete answer (the completion time), for each item.

  There is no easy way to determine the accuracy of these timings; although it can probably be assumed that they are accurate to within a couple of centiseconds when the subject's Web browser is running locally on a reasonably fast computer, the accuracy may be far lower if the browser is running as an X client across a network. However, the timings are useful for determining whether the subject took a break in the middle of the experiment, and may in some cases be accurate enough to indicate which experimental items were the causes of particular difficulty in responding.
- **Response Time Checking.** WebExp allows the experimenter to provide upper and lower limits for both the onset time and the completion time. If a subject exceeds these limits for one or more items, WebExp saves the entire set of responses in a specially named file, thus allowing the experimenter to discard them if necessary.
- **Generation of Off-Line Questionnaires.** Apart from allowing the administration of experiments online (over the Web or in a laboratory), WebExp also generates off-line questionnaires from the same experimental materials. WebExp produces $\LaTeX$ code that can be typeset and printed out directly to obtain paper-based questionnaires.

1.1.2 Desirable Features

The following is a non-exhaustive list of additions that may be necessary to make the applet more generally useful.

- **User-Defined Experimental Designs.** In the ideal case, the design of an experiment should be specified by the experimenter, instead of being hard-wired into the WebExp applets. The experimenter should be able to design the layout of an experimental item, and different answer types should be available, such as textual answers, numeric answers, and multiple choice answers.

- **Automatic Management of Variants.** An experiment can consist of multiple variants (e.g., lexicalizations), and WebExp will make sure that each variant is administered to an equal number of subjects. This does not take into account, however, that some of the subjects might end up producing non-usable data, because they fail authentication, exceed the time limits, or fail to complete the full experiment, etc. Ideally, WebExp should recognize these cases and repeat the failed variant of the experiment, instead of just proceeding blindly to the next variant.

1.1.3 Known Problems

None at the moment.

1.2 Web-Based Experiments

This is a short discussion of the pros and cons of Web-based experiments. For more a more extensive treatment of Internet experimentation, see Hewson et al. (1996).

1.2.1 Pros of Web-Based Experiments

- Saves time for the experimenter and the subject.
- (Putative) higher response rate from subjects and no danger of subjects agreeing to do it and then not showing up.
- Questions are presented in order, and the subject has no chance to change their answer to a previous question after seeing a later one.
- Subjects cannot accidentally fail to respond to an experimental item (the software will not allow the subject to just press the return key).
- Response timing may prove useful to check that the experiment has been completed, or even for evaluation.
- Subject authentication is better than if one allows subjects to take a questionnaire home.

1.2.2 Cons of Web-Based Experiments

- Subjects are self-selecting. Conducting the experiment over the Web restricts the subjects to those who are competent at using Web browsers, which may be an unrepresentative sample.
- Subject authentication is not as good as if the experiment is done under laboratory conditions.

Chapter 2

Using and Customizing the Toolbox

2.1 Components of WebExp

WebExp consists of two Java components, namely a server application (which is run on the host Web server) and a client applet (which is run remotely by the subject's Web browser). These components rely on a number of text files (which specify the experimental materials, etc.) and, in the client's case, on an HTML file that contains `APPLET` and `PARAM` tags specifying details of how the client is run.

The remainder of this chapter gives the information you need to get an experiment up and running. Section 2.2 provides an installation guide; section 2.3 explains what you will need to provide in order to run an experiment, and section 2.4 gives details of how to invoke the server and client software.

2.2 Installation

In this and later sections we make the following assumptions:

- You are installing WebExp onto a UNIX or UNIX-like system.
- The directory that contains user-specific Web space on your server is called `public_html`. You should replace `public_html` with the appropriate name if your system differs.
- The full path to `public_html` is `$HOME/public_html` (you should amend this path as appropriate).

WebExp is provided in as a gzipped tar file. The following steps have to be carried out to install it on your local host:

1. Create in your home directory a directory called `public_html` (if you don't have one already). This directory should have world read and execute permissions (use the command `chmod 755 public_html` to ensure this).
2. Copy the file `web_exp.tar.gz` into `public_html` and change to this directory.
3. Extract the files from the archive using, for example:

```
zcat web_exp.tar.gz | tar xvBf -
```

4. Make sure that all the newly expanded files and directories are executable for all users. If they aren't, then execute the following command:

```
chmod -R 755 *.class
```

2.3 Designing an Experiment

The WebExp client consists of a Java applet which is downloaded from the Web server running the WebExp server application. To run an experiment, you will need to create an HTML page that specifies the location of this applet and its parameters, as well as one or more files of experimental stimuli. This section contains information on how to achieve this.

2.3.1 Experimental Paradigms

Currently, WebExp supports two experimental paradigms:

- **Magnitude Estimation.** This paradigm can be used for experiments involving numeric judgments, for example of linguistic stimuli. After having judged a modulus item, the subject has to provide a judgment for each experimental item relative to the modulus. This paradigm also includes a calibration phase (judgments of line lengths) and a training phase. The experimental set-up was designed according to the proposals by Bard et al. (1996) and Cowart (1997) for studies on linguistic acceptability.

- **Sentence Completion.** In this paradigm, the subject is presented with an experimental item, and has to provide a textual answer. This paradigm can be used for sentence completion studies, for instance: the subject sees a sentence fragment and has to key in a word or phrase that completes the fragment to form a grammatical sentence. Of course the same set-up can be used for free-choice questionnaires in general.

Future Versions of WebExp might contain additional applet classes to support more experimental paradigms. If you would like to get involved in the development of new paradigms, please contact the WebExp developers to obtain the source code (see chapter 3).

2.3.2 Randomization

WebExp presents the items in the stimulus file (or files, if there are several variants) in an individual randomized order for each subject taking the experiment. The randomizer generates these orderings by randomly interleaving the items of each block in the stimulus file. Different constraints on randomization can be imposed. In the present implementation, the following types of randomization are supported:

- **Free Randomization.** The experimental items can appear in any order, blocking doesn't play a role.

- **Constrained Randomization.** The experimenter specifies restrictions on how many items from which block can be displayed consecutively.

- **Blocked Randomization.** The order of the blocks is randomized, while the items within each block are presented in a fixed order.

Which type of randomization is used in generating the stimulus lists is defined via the parameter `Randomisation Type` (see section 2.3.7 for details).

2.3.3 Stimulus Files

WebExp reads the experimental items from a stimulus file, which is a plain ASCII text file. The stimulus file has to have the following structure:

- The first item in the file is the reference sentence, preceded by a line containing the marker `@r`. (The reference sentence also has to be present if a completion paradigm is used, although it is ignored in this case.)
- The reference sentence is followed by an arbitrary number of blocks with experimental sentences, where each block is preceded by a code of the form `@c`n and enclosed in curly brackets. Here n is the block number. Each block consists of an arbitrary number of items. Each item has to be on a separate line, preceded by a line containing an item identifier.
- The item blocks are followed by a block of fillers, preceded by the code `@f` and enclosed in curly brackets. Each item in the filler block has to be on a separate line, preceded by a line containing an item identifier. Note that the filler block may only be present if the design type `blocked` (see section 2.3.7) is chosen. Otherwise, fillers are to be specified as ordinary blocks preceded by `@c`.

Arbitrary lines in the stimulus file can be marked as comments by prefixing them with `#`.

The block structure of the stimulus file is typically used to separate experimental items that are of different types (i.e., represent separate experimental conditions). The block structure also crucial to the way a randomization is computed for the stimulus file (see section 2.3.6 for details).

An example for a stimulus file is the following:

```
# reference
@r
Who did you find out if your brother met?

# stimuli for first condition
@c1
{
01.01
Which postman did you say when the only friend has met?
01.02
Which mother did you notice was persuaded to leave?
}

# stimuli for first condition
@c2
{
02.01
Which man has John taken a very blurry photograph of?
02.02
Which friend has Mary painted a very abstract picture about?
}

# fillers
@f{
F.01
```

```
Which postman did you assume this women saw?
F.02
Which song bird did you forget that Tom had killed?
F.03
Which mother old did you know if the friend persuaded?
F.04
What did you remind Tom his only friend convinced him of?
}
```

2.3.4 Web Pages

The Java applet that controls a WebExp experiment relies on two Web pages (the WebExp distribution contains examples of both pages):

- **Instructions Page:** this page is intended to contain general information on the experiment, plus the instructions that the subject needs to carry out the experiment.[1] The instructions should also explain how to fill in the personal details questionnaire that the subject has to complete before proceeding to the experiment proper.[2] The instructions page also contains HTML commands that interface with the Java applet that runs the experiment. These commands is used for passing parameters to the applet, as detailed in section 2.3.6.

- **Completion Page:** this is the page that is invoked after the subject has successfully completed the experiment. It is meant to contain a thank you note, and, where appropriate, further information on the experiment.

Note that it might sometimes be necessary to use a frame version of the instructions page (an example is provided as part of the WebExp distribution). This is necessary on some systems to get around a bug in Netscape: Some versions of Netscape stop a Java applet if the Web page that contains the applet is resized. This problem does not seem to occur if the applet runs inside a frame.

2.3.5 Other Files

You will also need to create a preformated e-mail message which is sent to subjects on completion of the experiment. This e-mail should normally contain contact details for the project team, and (if appropriate) debriefing information or a pointer to debriefing information on the World Wide Web.

[1] It is important to include the credentials of the experimenter here (affiliation, type of project, funding source, etc.), as well as to provide the possibility for subjects to supply feedback, as pointed out by Hewson et al. (1996).

[2] On the issue of administering experiments using WebExp, Antonella Sorace (personal communication, 1998) points out that the magnitude estimation paradigm tends to have low face validity, particularly with non-mathematically minded subjects. It is also not uncommon for subjects to misunderstand the nature of the task, especially the concept of proportionality. In a face-to-face experiment it is normally possible for the experimenter to intervene and provide further clarification, or rectify subjects' mistakes if they are apparent. In a web-based experiment, on the other hand, this wouldn't be possible. It is therefore crucial to ensure that the instructions are absolutely clear, and that the training phase is effective. It might be appropriate to include examples of what **not** to do in the training phase, such as for instance using a 10-point scale—a tendency that subjects often display. It is possible, however, that most subjects tested through WebExp would be less likely to have these problems since, they would be a self-selecting group of computer-literate people.

2.3.6 Applet Parameters

This sections provides some information on how to customize WebExp for a given experimental design. The commands detailed here should be included in the Instructions Page outlined above.

The Web page of the experiment contains an <APPLET> ... </APPLET> tag which specifies a set of parameters to be passed to the client applet that administers the experiment:

1. <APPLET code="*class_file*" width=x height=y>

   This tag gives the name of the class file that contains the applet, as well as the width and height of the display area for the applet. Two values for *class_file* are supported by the present version of WebExp: experiment.MagestApplet.class for magnitude estimation experiments, and experiment.CompletionApplet.class for completion studies (details of these paradigms are in section 2.3).

2. <PARAM name="Mail Host" value="*hostname*">

   Here *hostname* specifies the name of the mail host that the applet attempts to connect to to send mail to the subject. Make sure to specify a host that accepts SMTP connections from the machine the WebExp server application runs on, else a security violation will occur.

3. <PARAM name="Return Email" value="*email_address*">

   After completing the experiments, an e-mail will be sent to the subject in order to check if his/her e-mail address is valid. The address specified by this parameter will be used as the sender's address.

4. <PARAM name="Return Name" value="*string*">

   This parameter contains the name of the experimenter. This is used for sending e-mail messages to the subject once he/she has completed the experiment.

5. <PARAM name="Return Subject" value="*string*">

   This parameter specifies the subject of the e-mail that is sent to the subject once he/she has completed the experiment.

6. <PARAM name="Experiment Name" value="*string*">

   The name of the experiment can be specified with this parameter.

7. <PARAM name="URL" value="*url*">

   This parameter allows to specify the HTTP address of the server from which the applet is downloaded. This information is used for additional verification of the identity of the subject.

8. <PARAM name="Input Directory" value="*directory*">

   This parameter allows to set the directory where the input files can be found (the stimulus files and the file containing the e-mail addresses already used). The entire path must be specified (including the home directory).

9. `<PARAM name="Output Directory" value="`*directory*`">`

   This parameter allows to set the directory where the result files are put. The entire path must be specified (including the home directory).

10. `<PARAM name="Variants" value="`n`">`

    This parameter specifies the number of variants of the current experiment. Variant are different versions of the same experiment that are to be administered to differed subjects. The use of variants allows to conduct experiment with a between subject design (an example are different lexicalizations used in a psycholinguistic study).

    WebExp administers the variants using the following system: the first variant is administered to the first subject, the second one to the second subject, etc. If the n, the maximum number of variants is reached, then WebExp administers the first variant again, then the second one, etc.

11. `<PARAM name="Port" value="`*port_no*`">`

    This parameter contains the number of the port on which the server application runs. The port value has to be identical to the port number specified with the server application (see section 2.4.1).

12. `<PARAM name="Stimulus File" value="`*filename*`">`

    This parameter specifies the name of the file containing the experimental stimuli. If there are several variants of the experiment, only the first part of the file names of the variants has to be specified. A suffix containing the variant number will be generated automatically for each variant.

13. `<PARAM name="Template File" value="`*filename*`">`

    This parameter is used for generating a LaTeX version of the experimental items. The *filename* contains a template for the LaTeX file from which paper-based questionnaire can be generated. Please refer to the example template included in the WebExp distribution for details.

14. `<PARAM name="Thanks Message" value="`*filename*`">`

    This parameter specifies the name of the file containing the e-mail message to be sent to the subject once he/she has completed the experiment.

15. `<PARAM name="Thanks URL" value="`*url*`">`

    This parameter specifies the Completion Page, i.e., the Web page to be displayed once the subject has completed the experiment.

16. `<PARAM name="Randomisation Type" value="free|constrained|blocked">`

    This parameter has to have one of three values: `free`, `constrained`, or `blocked`. It specifies the randomization type used to display the sentences in the stimulus files. If the parameter is left out, the default value is `free`. If the value is `constrained` then the value of the `Randomisation Args` parameter has to be specified also.

17. `<PARAM name="Randomisation Args" value="`*rand_args*`">`

    Here the constraints on randomization have to be specified in case the randomization type `constrained` is used. See section 2.3.7 for more detail.

18. `<PARAM name="Minimum Onset Time" value="`n`">`

    This parameter specifies the minimum time (in milliseconds) that may elapse between the display of the experimental item and the first keystroke of the subject (the onset time). This timing is used to make sure that the subject completes the experiment correctly. If the onset time for one or more of the experimental items is smaller than this threshold, then the answer of the subject will be saved in a junk file.

19. `<PARAM name="Maximum Onset Time" value="`n`">`

    This parameter specifies the maximum time (in milliseconds) that may elapse between the display of the experimental item and the first keystroke of the subject (the onset time). If the onset time for one or more of the experimental items is larger than this threshold, then the answers of the subject will be saved in a junk file.

20. `<PARAM name="Minimum Completion Time" value="`n`">`

    This parameter specifies the minimum time (in milliseconds) that may elapse between the onset (the first keystroke of the subject) and the completion of the item (the pressing of the return key). If the completion time for one or more of the experimental items is smaller than this threshold, then the answer of the subject will be saved in a junk file.

21. `<PARAM name="Maximum Completion Time" value="`n`">`

    This parameter specifies the maximum time (in milliseconds) that may elapse between the onset (the first keystroke of the subject) and the completion of the item (the pressing of the return key). If the completion time for one or more of the experimental items is larger than this threshold, then the answer of the subject will be saved in a junk file.

22. `<PARAM name="Email Checking" value="yes|no">`

    This parameter determines whether or not the e-mail address the subject provides will be checked against a database containing the e-mails of subject who have already taken the experiment. If enabled, this feature should prevent subjects from taking the experiment repeatedly.

23. `<PARAM name="Language" value="`*lang_code*`">`

    This parameter identifies the language used in the experiment's user interface. The languages codes currently supported are `EN` for English and `DE` for German.

24. `<ALT="`*string*`">`

    This tag allows the experimenter to specify a message to be displayed in case the subject's Web browser is not capable (or not configured) to run Java.

25. `</APPLET>`

    This tag indicates the end of the applet specification.

2.3.7 Randomization Parameters

This section gives more detail on the randomization. Note that the type of randomization used by WebExp depends on the value passed to the parameter `Randomization Type`.

2.3.7.1 Free Randomization

The option `free` means that no restrictions are imposed on the randomization. The WebExp randomizer will select the items in the stimulus file in random order, without taking the block structure of the stimulus file into account.

2.3.7.2 Constrained Randomization

If the option `constrained` is chosen, then WebExp expects a parameter `Randomisation Args` to be present. The value of this parameter is a set of randomization constraints. The following syntax has to be used:

$n_1 \texttt{ of } b_{11}\texttt{, } b_{12}\texttt{, } b_{13}\texttt{, } \ldots\texttt{, } b_{1k_1}\texttt{;}$
$n_2 \texttt{ of } b_{21}\texttt{, } b_{22}\texttt{, } b_{23}\texttt{, } \ldots\texttt{, } b_{2k_2}\texttt{;}$
$\ldots\texttt{;}$
$n_m \texttt{ of } b_{m1}\texttt{, } b_{m2}\texttt{, } b_{m3}\texttt{, } \ldots\texttt{, } b_{mk_m}$

Here, n_1 to n_m and b_{11} to b_{mk_m} are placeholders for numbers. The meaning of a constraint of the form n **of** b_1**,** b_2**,** b_3 is that at most n items from the blocks b_1, b_2, and b_3 are allowed to occurs consecutively. Randomization constraints are therefore a means of avoiding long sequences of items from the same conditions (i.e., from the same block), as well as long sequences of fillers.

Please note the following restrictions:

- The number of the blocks in the stimulus file (without the filler block) must be a multiple of $n_1 + n_2 + \ldots + n_m$.
- Each n_i must be at most the number of sentences per block, b_{ij_i}.
- Each b_{ij} must be at most the maximum number of blocks in the stimulus file.

All randomization constraints must be written on the same row (in the applet parameter). As an example, consider the following valid set of constraints:

```
<param name="Randomisation Type" value="constrained">
<param name="Randomisation Args" value="3 of 1,2,3,4; 1 of 1,3; 1 of 2,4">
```

2.3.7.3 Blocked Randomization

The `blocked` randomization option caters for the following type of experimental design:

- All items in a block appear in a fixed order, viz., in the order specified in the stimulus file.
- The blocks as a whole appear in random order.
- Each block is followed by a set of fillers, where the number of fillers is equal to the number of items in the block.

The stimulus file has to contain a designated filler block (at the end of the file, preceded by the code `@f`). The number of fillers in the filler block has to be equal to the total number of items in the experimental blocks. The fillers for each experimental block are selected randomly from the filler block.

2.4 Running an Experiment

This section describes the steps that are necessary to run a WebExp experiment, once installation and experimental design are complete.

2.4.1 Server Application

The server application runs on the Web server that hosts the experiment, and waits for client applets to connect to it. It issues experimental materials to clients, and records subjects' responses.

Note that the server application has to be run on the machine from which the client applet is downloaded, i.e., on the Web server which hosts the Web page of the experiment. Otherwise, a security violation will ensue and the client applet will not run.

To run the server application execute the following steps (while logged on to the Web server from which the experiment is to be run):

1. Change directory to `$HOME/public_html/web_exp/server/`.
2. Check if your `CLASSPATH` variable is set to `.:$HOME/public_html/web_exp`. If it isn't, then set the it using the following commands (using bsh and csh, respectively):

   ```
   export CLASSPATH = .:$HOME/public_html/web_exp
   ```

   or:

   ```
   setenv CLASSPATH .:$HOME/public_html/web_exp
   ```

3. Now start the server application:

   ```
   java ClSv [-pport_no]
   ```

   The *port_no* value must be the same as the value of the port parameter of the client applet. This value is specified in the Web page of the experiment (see section 2.3.6). If several instances of the server application are being run (to administer several experiments concurrently), then it is necessary to specify a distinct port value for each of them, both in the command line of the server application and on the Web page of the experiment. If no port value is given, both the server and the client default to port 6789.

After completing all these steps it is possible to run a client application from any computer connected to the Web. To terminate the server application, press `Ctrl-C`.

2.4.2 Client Applet

The client applet is run by pointing a Web browser at the Instructions Page (see section 2.3.4). The following steps are required:

1. If you are using UNIX, make sure that the `CLASSPATH` variable of your environment is set to null before starting your Web browser. If it isn't, then execute (using bsh and csh, respectively):

   ```
   export CLASSPATH =
   ```

   or:

   ```
   setenv CLASSPATH
   ```

2. Now start your Web browser and load the Instructions Page, which embeds the client applet. Note that the Web browser has to be Java-compatible, and Java has to be activated.

Important: Note that the client applet attempts to make a TCP/IP connection through an unregistered port to the Web server running the server application. This means that the machine running the client has to allow such a connection, which might not be the case if this machine accesses the Net through a firewall or proxy server which is heavily secure (normally, firewalls allow the *initiation* of connections through unregistered ports by their own machines).

Unfortunately, the client applet will not run in such a case. (One possible workaround might be to run WebExp via the port of an existing Internet service that is allowed through the firewall. However, this is not likely to be a very robust strategy, and we would strongly discourage its adoption on the grounds that firewalled service providers may well interpret it as a security risk.)

2.5 Response Files

At the end of the experiment, the client applet establishes an Internet connection to the WebExp server. It transmits the responses the subject has provided during the experiment, and the server stores them in a set of files. These files are located in the directory specified by the `Output Directory` parameter (see section 2.3.6).

Furthermore, a "thank you" message is sent to the subject (at the address provided in the personal details questionnaire). This message will bounce back to the experimenter if the subject enters a bogus e-mail address, and the experimenter can then exclude this data from evaluation.

WebExp ggenerates the following response files for an experiment that uses the class `MagestApplet` (here, *filename* is the value of the parameter `Stimulus File` and n is the number of the current subject):

- **Data Files:** the following files contain the responses of the subject for the calibration task, the practice task, and the main experimental task: `data_`*filename*`.calib.`n, `data_`*filename*`.pract.`n, and `data_`*filename*`.main.`n.

- **Junk Files:** in case subject authentication fails, or the time limits for responding to one or more experimental items are exceeded, WebExp creates so-called junk files instead of data files: `junk_`*filename*`.calib.`*n*, `junk_`*filename*`.pract.`*n*, and `junk_`*filename*`.main.`*n*. The experimenter can then decide post-hoc whether to include these data into the final evaluation or not.

- **Summary Files:** to make data evaluation easier WebExp also generates summary files that contain the combined contents of all data files (but not of the junk files): `data_`*filename*`.calib.all`, `data_`*filename*`.pract.all`, and `data_`*filename*`.main.all`.
- **Questionnaire Files:** finally, WebExp also produces LaTeX files for generating a questionnaire containing the experimental materials. These files are useful for conducting a validation study of a Web-based experiment: `form_`*filename*`.calib.`*n*`.tex`, `form_`*filename*`.pract.`*n*`.tex`, and `form_`*filename*`.main.`*n*`.tex`.

For an experiment that the `CompletionApplet` class, the following files are created: `data_`*filename*`.`*n*, `junk_`*filename*`.`*n*, `data_`*filename*`.all`, and `form_`*filename*`.`*n*`.tex`.

Chapter 3

Copyright Notice

This chapter contains some information on copyright issues relating to WebExp, a set of Java classes for Web-based psychological experiments (hereafter called Classes).

Please note: The information given here should be considered as an informal statement by the people involved in the development of the Classes. We do not intend to make any legally binding statements on the issues raised (nor are we in a position to do so).

1. The copyright for the Classes rests with its developers. The current developers are:
   (a) at the University of Edinburgh: Centre for Cognitive Science, Human Communication Research Centre, and Department of Psychology (Martin Corley, Steffan Corley, Matt Crocker, Frank Keller);
   (b) at the University of the Saarland: Department of Computational Linguistics and German Research Center for Artificial Intelligence (Lars Konieczny, Amalia Todirascu[1]).
2. The Classes are distributed free of charge for non-commercial purposes.
3. The Classes are distributed in the hope that they will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose.
4. The Classes are distributed in the form of byte code only, and re-distribution by third parties is not permitted. Please make sure that you obtain the latest version of the Classes by contacting the developers.
5. If you would like to extend or modify the Classes, please contact the developers. Source code will be made available provided that you are willing to return any modified sources or new classes to the development team. We hope that in this way the Classes can be maintained and improved in a shared development effort.
6. The use of the Classes for conducting psychological experiments is allowed under the following conditions:
   (a) The developers are acknowledged in any questionnaires and experiments conducted using the Classes, as well as in any research and publications derived from these experiments.

[1]Amalia Todirascu took part in the development of WebExp while she was a visiting researcher at DFKI. She is now at the Department of Computer Science, Al. I. Cuza University of Iasi.

(b) Any experiments that make use of the Classes are advertised via a link from the Central Entry Point for Web-based experiments, currently located at `http://surf.to/experiments/`. This restriction is meant to ensure a shared subject pool for Web-based experiments. To have your experiment added to the Central Entry Point pages, please contact the developers.

7. The current contact address for the developers is:

Martin Corley
Department of Psychology
University of Edinburgh
7 George Square
Edinburgh EH8 9JZ, UK

Please subscribe to the following mailing list if you would like to receive announcements concerning the WebExp Classes:

`webexp@cogsci.ed.ac.uk`

To subscribe to this list, please send a message containing the word 'subscribe' in the message body to:

`webexp-request@cogsci.ed.ac.uk`

If you would like to obtain the latest version of the Classes (or the source code, in case you want to participate in the development), please contact the following address:

`webexp-dev@cogsci.ed.ac.uk`

This address should also be used for any other queries regarding the Classes, including bug reports.

Bibliography

Bard, Ellen Gurman, Dan Robertson, and Antonella Sorace. 1996. Magnitude Estimation of Linguistic Acceptability. *Language* 72(1): 32–68.

Cowart, Wayne. 1997. *Experimental Syntax: Applying Objective Methods to Sentence Judgments*. Thousand Oaks, CA: Sage Publications.

Hewson, Claire M., Dianna Laurent, and Carl M. Vogel. 1996. Proper Methodologies for Psychological and Sociological Studies Conducted via the Internet. *Behavior Research Methods, Instruments, and Computers* 28: 186–191.