|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [User Guide](http://www.opendial-toolkit.net/user-manual)‎ > ‎  Getting started   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **1. Downloading OpenDial**  The [Download](http://www.opendial-toolkit.net/download) page contains the various OpenDial releases.  **System requirements**:   * [**Java 8**](https://www.java.com/en/download/help/index_installing.xml) * [Gradle](http://gradle.org/) (if you want to modify or extend the source code)   NB: If you are using Maven/Gradle for your software and are only interested in fetching OpenDial as an external dependency, there a Maven package for Opendial on [Jcenter](https://bintray.com/plison/maven/OpenDial) (groupId:opendial,artifactId:opendial).  **2. Running OpenDial**  When you have downloaded the package, simply unzip it and cd to the main directory:  unzip opendial-X.zip  cd opendial  (of course, X should be replaced by the particular version you are using).  If you have downloaded one of the stable packages, the source code is already compiled, so you can directly run OpenDial out-of-the-box by executing one of the following scripts:   |  |  | | --- | --- | | **- Linux / Mac OS X:** | ./scripts/opendial | | **- Windows:** | .\scripts\opendial.bat |   If everything goes well, you should see a window like this pop up:  [http://www.opendial-toolkit.net/_/rsrc/1437063539059/user-manual/getting-started/opendial_start.png](http://www.opendial-toolkit.net/user-manual/getting-started/opendial_start.png?attredirects=0)  If you get an  "UnsupportedClassVersionError" exception instead, this probably means that your version of Java is older than Java 8 (you can check this by typing java -version).  This means you should update your Java installation to the latest version. Simply google "*how to install Java 8 on [your platform]*" to find the easiest way to do so (ready-made packages are available for most platforms). If you experience other types of problems running OpenDial, do not hesitate to send an email to the [opendial-users](http://www.opendial-toolkit.net/contact) mailing list.  From there, you can either start a new dialogue domain (Domain > New) or load one of the domain examples in the directory domains/examples (Domain > Open File).  **3. Compiling OpenDial from source (*optional*)**  In case you plan to work on OpenDial's source code, or if you have fetched the bleeding-edge source code from the github repository, you will need to recompile the source. Since version 1.4, OpenDial relies on the [Gradle](http://gradle.org/) build framework (before 1.4, we used [ant](https://ant.apache.org/)).  If you do not yet have Gradle on your machine, you first need to install it (on Mac OS X, you can install it via [Brew](http://brew.sh/), and a gradle  package is also available on Ubuntu).  Once this is done, simply go to the main directory and type:  gradle compile  This should compile the source code in a few seconds. You can verify that everything works correctly by running the unit tests for Opendial:  gradle test  Once the compilation is completed, you can run OpenDial through the scripts mentioned above (./scripts/opendial or .\scripts\opendial.bat).  **4. Using OpenDial**  You can try out OpenDial by opening a dialogue domain such as the one specified in [domains/example/example-step-by-step\_fixed.xml](https://github.com/plison/opendial/blob/master/domains/examples/example-step-by-step_fixed.xml) and typing a new user input such as: "please turn left". You should then see in the chat window the system answer: "OK, turning Left".  OpenDial contains 3 tabs: an**interaction**tab listing the dialogue history, a **state monitor**tab showing the current dialogue state and its latest updates, and a **domain edito***r* tab allowing you to edit the domain specification.  *Interaction tab*  The first tab displays the dialogue history and allows the user to interactively test the dialogue system.  [http://www.opendial-toolkit.net/_/rsrc/1437067038241/user-manual/getting-started/opendial_chat.png](http://www.opendial-toolkit.net/user-manual/getting-started/opendial_chat.png?attredirects=0)  At the bottom of the page, a text field can be used to enter new inputs (this feature is useful to test the system behaviour without a speech recognizer). N-best lists can be entered by specifying probabilities in parentheses and separating alternative hypotheses with a semicolon (;). For instance, the user input:  now move left (0.55) ; do not move left (0.15)  represents a N-best list with three alternative elements: "*now move left*" with probability 0.55, "*do not move left*" with probability 0.15, and a last empty element (no input) with probability 0.3.  If you include modules related to [speech recognition or synthesis](http://www.opendial-toolkit.net/user-manual/speech-recognition-and-synthesis)to your architecture, you will also see a bottom panel allowing you to easily provide speech inputs.  *State Monitor Tab*  The state viewer displays the current dialogue state as a Bayesian network.  [http://www.opendial-toolkit.net/_/rsrc/1437067682687/user-manual/getting-started/opendial_statemonitor.png](http://www.opendial-toolkit.net/user-manual/getting-started/opendial_statemonitor.png?attredirects=0)  The user can click on each variable (or sets of variables) and derive their marginal probability distribution. In addition, the distribution of individual variables can be represented graphically in a chart or density function.  In addition to the current dialogue state, the system can also keep track and record past dialogue states. This can be very useful for debugging, as it shows how the dialogue state evolves during processing. The menu item Options > Record Intermediate States allows you to determine which states to record.  *Domain Editor Tab*  Finally, the domain editor allows you to interactively edit the dialogue domain (in XML format).  [http://www.opendial-toolkit.net/_/rsrc/1437067948973/user-manual/getting-started/opendial_editor.png](http://www.opendial-toolkit.net/user-manual/getting-started/opendial_editor.png?attredirects=0)  The editor is a XML editor with syntax highlighting and auto-completion features (for instance, if you type <rule>, the editor will automatically generate for you a basic skeleton for your rule specification). When you have made changes to the domain, you can save it (Domain > Save or the usual keyboard shortcut for your platform). This will both save the dialogue domain on disk, and update the dialogue domain in OpenDial.   **5. Command line parameters**  As mentioned above, you can run OpenDial through the scripts available in the scripts directory. Alternatively, you can also directly run the Java main class: java -cp "lib/\*" opendial.DialogueSystem.    You can specify additional (optional) parameters at runtime via the command line:  ./scripts/opendial -Dname\_of\_param1=value1 -Dname\_of\_param2=value2 ...  where name\_of\_paramX is one of these parameters:   |  |  | | --- | --- | | **Parameter name** | **Value** | | domain | File path for domain specification (cf. [Dialogue Domains](http://www.opendial-toolkit.net/user-manual/dialogue-domains)) | | dialogue | File path of previously recorded dialogue to import | | simulator | Filepath of domain specification for a user simulator (cf. [Parameter Estimation with Simulated Dialogues](http://www.opendial-toolkit.net/user-manual/parameter-estimation#With_simulated_dialogues)) | | gui | true/false to activate or deactivate the graphical interface (default is true). |   Please note the -D standing before each parameter name (these parameters are JVM arguments).  **6. Menu bar**  The OpenDial menu bar offers several functions:   |  |  | | --- | --- | | *Domain* > *New* | Creates a new, empty dialogue domain | | *Domain > Open File* | Loads an existing dialogue domain | | *Domain > Save* | Save the text in the domain editor | | *Domain > Save As* | Save the domain in another file | |  |  | |  |  | | *Domain* > *Import* > *Dialogue State* | Imports a previously saved dialogue state into the system | | *Domain* > *Import* > *Parameters* | Imports parameter variables into the system | | *Domain* > *Export* > *Dialogue State* | Exports the current dialogue state into an XML file | | *Domain* > *Import* > *Parameters* | Exports the current parameters into an XML file | | *Interaction > Reset* | Resets the interaction (set the state to the initial dialogue state for the domain) | | *Interaction* > *Pause/Resume* | Pauses or resumes the current interaction | | *Interaction* > *Connect to Remote Client* | Connects to a remote machine that also runs OpenDial (in order to e.g. conduct Wizard-of-Oz experiments) | | *Interaction* > *Interaction role* | If the system is connected to a remote client, selects the role (user or system) played by the local machine. | | *Interaction* > *Import Dialogue From* | Imports a previous saved dialogue and runs through it again | | *Interaction* > *Save Dialogue As* | Exports the dialogue history to an XML file | | *Options* > *View Utterances* | Determines how to display the user utterances in the chat window | | *Options* > *Record Intermediate States* | Determines whether to record intermediate states in the state viewer | | *Options* > *Show/Hide Parameters* | Determines whether to display parameters in the state viewer | | *Options > Settings* | Changes the current system settings | | |