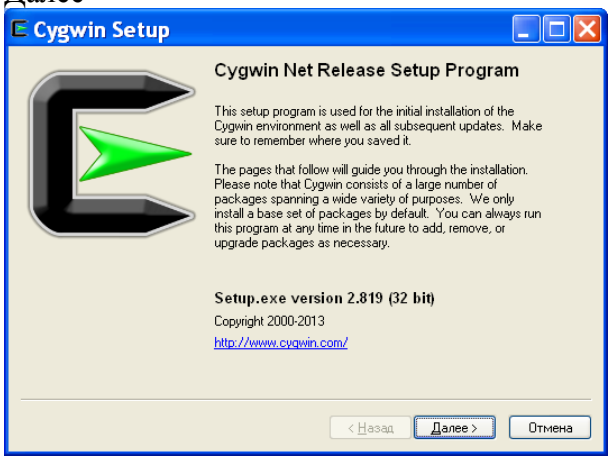
Настройка инструментов и окружения

Windows :: cygwin – эмулятор UNIX

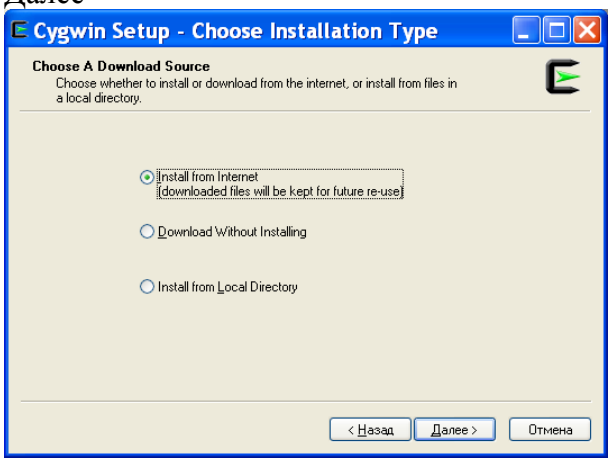
- *режим командной строки* («консольный»): удобная для программирования парадигма
- *эмулятор UNIX-системы*: большинство команд для работы с файловой системой, сетью, инструменты программиста, администратора и прочие утилиты доступны для использования
- *пакетная система расширений*: дополнительные утилиты ставятся отдельно, по-желанию, пакетным менеджером встроенным в `cygwin.exe` (или сторонним `apt-cyg`'ом)
- удобен для использования вместе с *python*
- *требования:* основные навыки работы с консолью UNIX

Установка

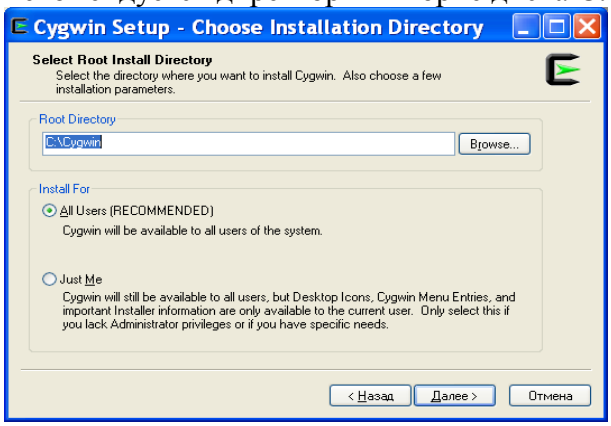
1. Скачать и запустить установщик.
2. Далее



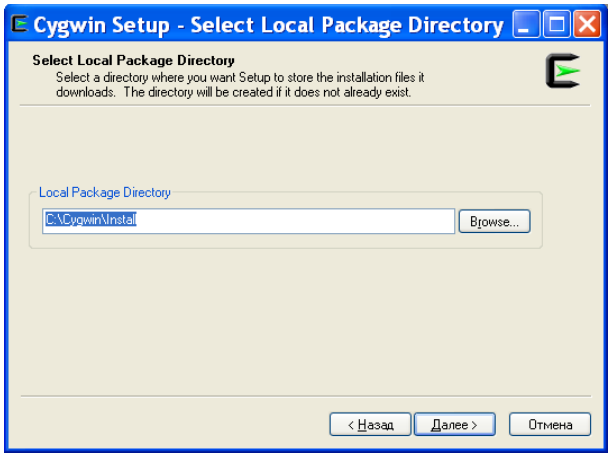
3. Далее



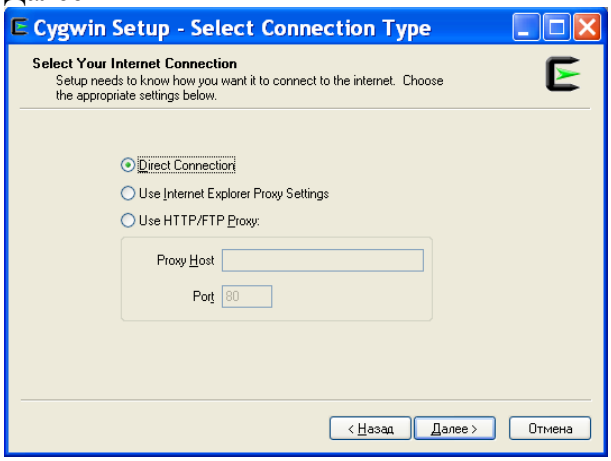
4. Рекомендуется директория в корне диска C. Далее



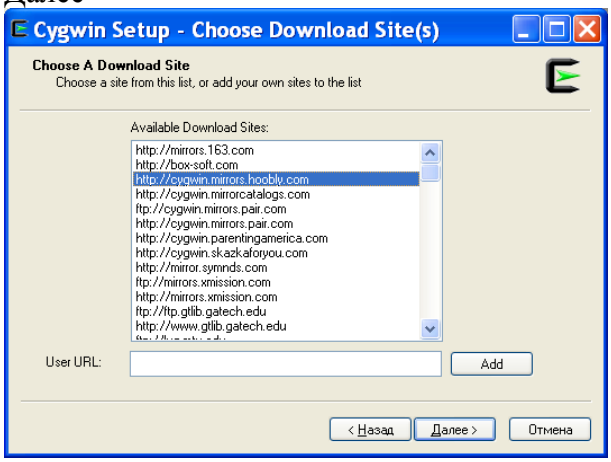
5. Место для кеша файлов устанавливаемых пакетов. Например, может быть в самой папке cygwin. Далее



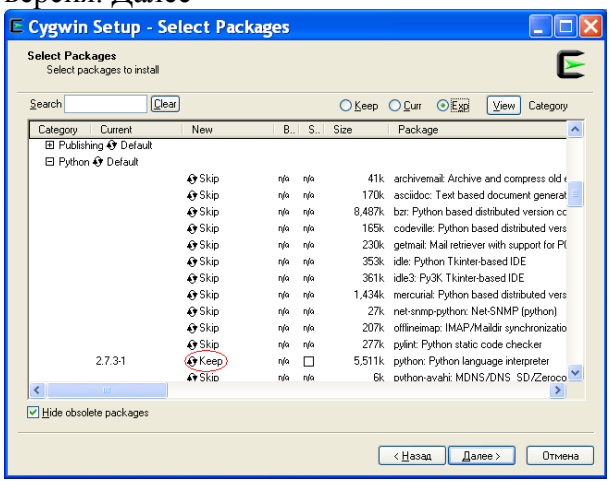
6. Далее



7. Далее



8. Выбрать категорию пакетов *Python*, найти пакет под именем *python* (**не** *python3***!**), кликнуть в красном кружке (см. скриншот) столько раз, чтобы там отображалась самая последняя версия. Далее

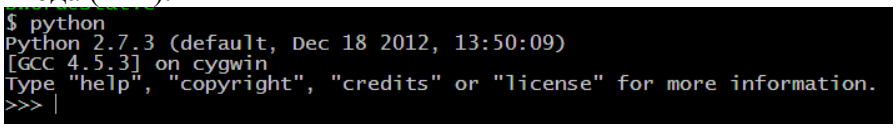


9. Подождать конца и готово!

Работа с интерпретатором

Запуск интерпретатора и выход из него

1. Запустить установленный cygwin.
2. Дождавшись черного окна; вбить команду *python* и нажать ENTER для входа в *интерактивный режим*.
3. Вот мы и запустили интерпретатор python! Весь код можно вводить после приглашения для ввода (>>>).



4. Для выхода можно:
   - закрыть окно – это прервет работу интерпретатора и закроет *cygwin*
   - нажать на клавиатуре сочетание клавиш, отвечающее за окончание ввода (EOF-маркер). В *cygwin*'е это CTRL+D
   - набрать инструкцию:
     ```
     import sys; sys.exit()
     ```
   - или:
     ```
     raise SystemExit()
     ```
   - или:
     ```
     exit()
     ```

Выполнение python-скриптов

- Для запуска в обычном режиме передадим имя файла скрипта в качестве аргумента интерпретатору. Такая команда выполнит все действия, записанные в скрипте в соответствии с правилами языка:
  ```
  python script.py
  ```
- Для запуска в *интерактивном режиме* выполним ту же команду без параметров:
  ```
  python
  ```
- Отличие от предыдущего способа запуска в том, что сначала пишется скрипт, а затем выполняется. А в интерактивном режиме выполнение происходит по мере написания: каждая набранная (прямо рядом с >>>) инструкция сразу выполняется и видно результат ее работы. Именно поэтому такой режим очень удобен для обучения языку.
- Чтобы запустить скрипт-файл из интерактивного режима, можно написать вот такую инструкцию прямо, после приглашения для ввода в виде трех знаков больше:
  ```
  >>> execfile('script.py')
  ```

Дополнительные инструменты

Менеджер пакетов

1. Python – расширяемый многочисленными модулями язык. Немудрено, что у него есть целый свой пакетный менеджер! На самом деле их много, но мы остановимся на самом используемом: **PIP**
   - знает о том, где взять, куда поставить, как настроить и как удалить указанный модуль для *python*
   - *очень* прост в использовании
   - стал почти де-факто менеджером пакетов для *python*
   - мониторит Python Packages Index, GitHub и другие места обитания пакетов
   - понадобиться нам для установки красивостей и инструментов отладки
2. Установим *pip*. Для этого, по совету pip-installer.org выполним в *cygwin*'е команду (утилита `curl` должна быть предварительно поставлена через `cygwin.exe`):
   ```
   curl -O https://raw.github.com/pypa/pip/master/contrib/get-pip.py
   ```
3. Мы скачали питонский скрипт, который установит нам *pip*, когда мы его выполним. Ради интереса можно посмотреть внутрь. Выполняем:
   ```
   python get-pip.py
   ```
4. Произойдет установка *pip*'а. `get-pip.py` более не нужен:
   ```
   rm get-pip.py
   ```

iPython

1. *pip* пригодиться еще не раз. Вот, например, уже сейчас, когда мы будем ставить продвинутую оболочку интерпретатора: **iPython**.
   - *python* – оригинальный интерпретатор; *ipython* – его оболочка, вносящая множество дополнительных функций и удобных мелочей
   - автодополнение по табу, цветной вывод, работа с консольными утилитами прямо из интерпретатора, история ввода и вывода, и еще много фишек
   - для *изучения* языка особенно удобно автодополнение
2. Выполним простую команду:
   ```
   pip install ipython
   ```
3. Готово! Вот так просто. Для входа в интерпретатор теперь используем *ipython*, вместо *python*. Не принципиально, какой командой пользоваться, при запуске скриптов: *ipython* – только украшающая надстройка над оригинальным *python*.

Написание скриптов

Текстовый редактор

- Так как скрипт – это просто текстовый файл, для написания кода можно пользоваться любым текстовым редактором.
- Но есть специальные текстовые редакторы для разработчиков, такие как Notepad++ и Sublime Text. К примеру, будем использовать второй вариант.
- Итого схема работы такая:
    1. пишем код в текстовом редакторе
    2. сохраняем в файл с расширением *.py*
    3. запускаем скрипт, вызывая интерпретатор в командной строке
    4. возвращаемся к первому пункту, при неудовлетворительной работе скрипта