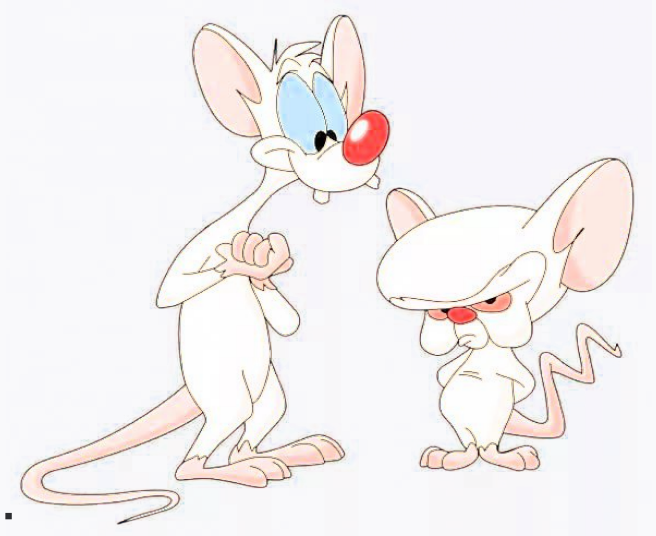
Клиент-серверные приложения на Python

Урок 1

Концепции хранения информации

Особенности хранения символов в памяти компьютера. Недостатки кодировки ASCII. Введение в кодировку Unicode. Unicode в Python 3. Конвертация байтов и строк. Примеры. Ошибки преобразования.

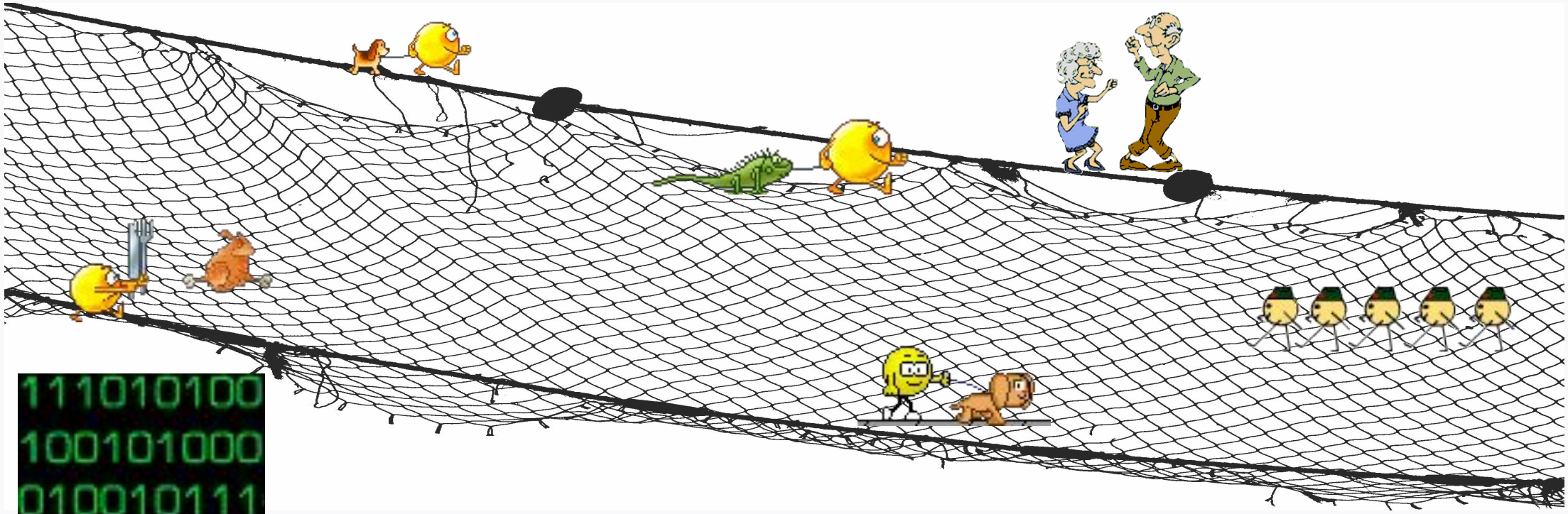
Цели урока



Понять:

- Как символы хранятся в памяти компьютера;
- Почему кодировки ASCII оказалось недостаточно;
- Что такое стандарт Unicode и как он реализован в Python 3;
- Как выполнять конвертацию данных между форматами.

По сети гуляют… байты, хотя мы видим символы



Кодировка ASCII

Недостатки:

- Работает только с 256 символами;
- Строгая привязка шрифтов;
- Сложность конвертации между форматами;
- Сложность использования на ПК с различными ОС;
- Невозможность использования с неалфавитными системами письма.

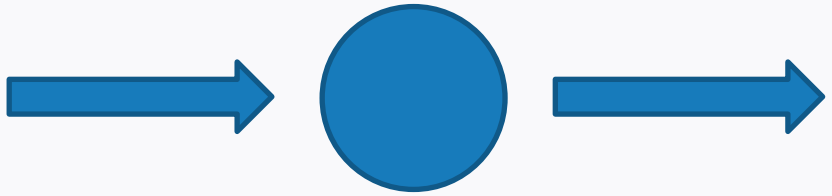
Стандарт Unicode

Для чего он нужен?

Отделяет символ от его представления в памяти компьютера и отображения на устройстве вывода

Хранение символа в компьютере

Кодовая точка Unicode



Например, U+041F

Отображение символа на мониторе

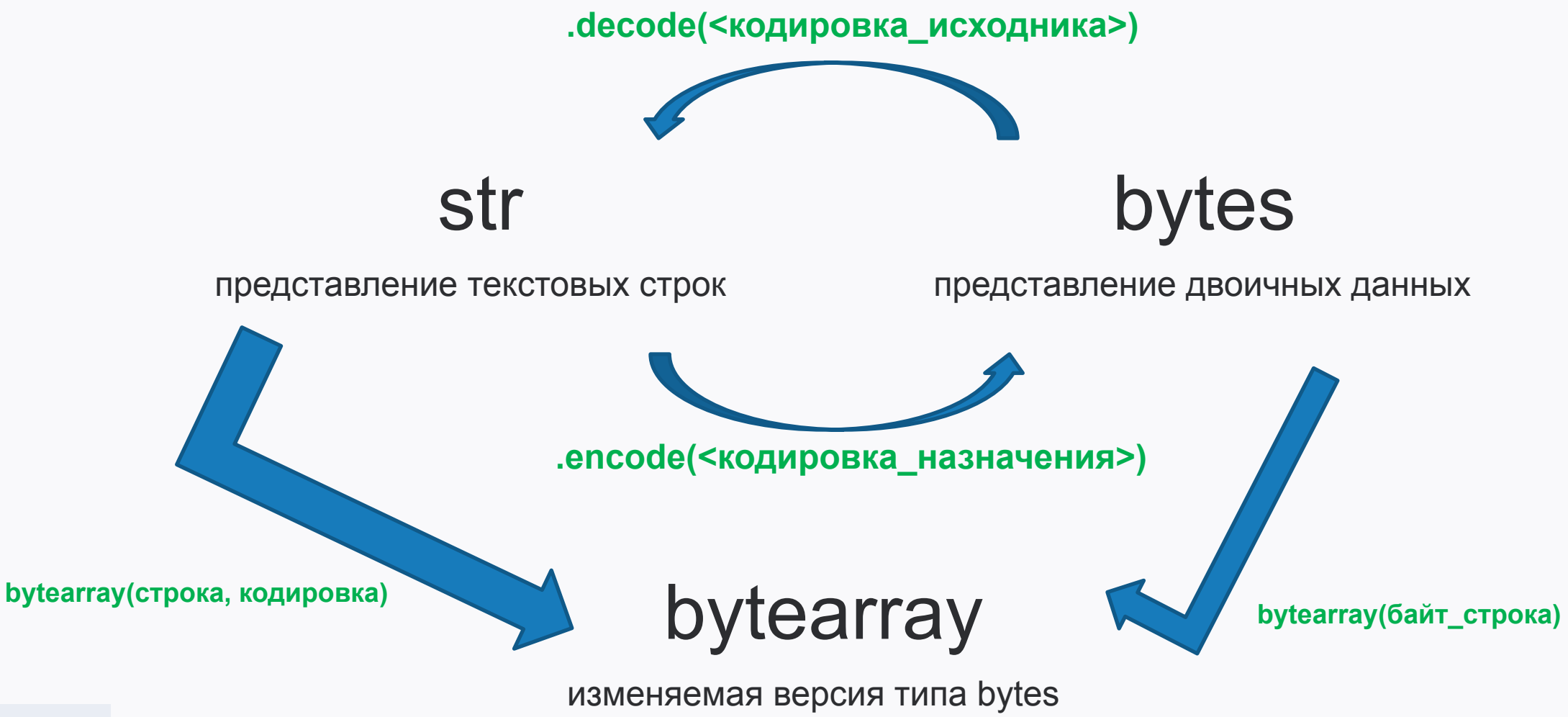
Кодирование кодовых позиций

Для этого в Unicode применяются кодировки UTF-8 и UTF-16

UTF-8

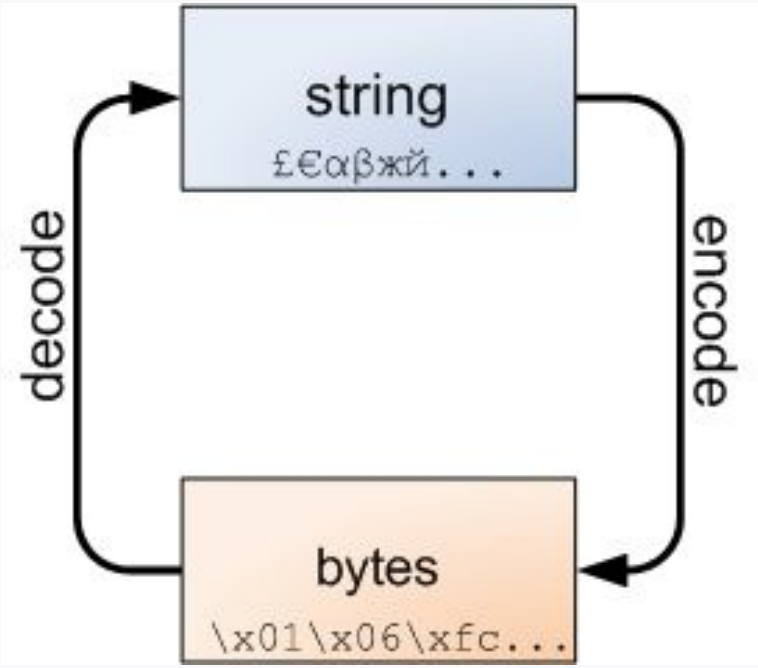
UTF-16

Python 3 и Unicode



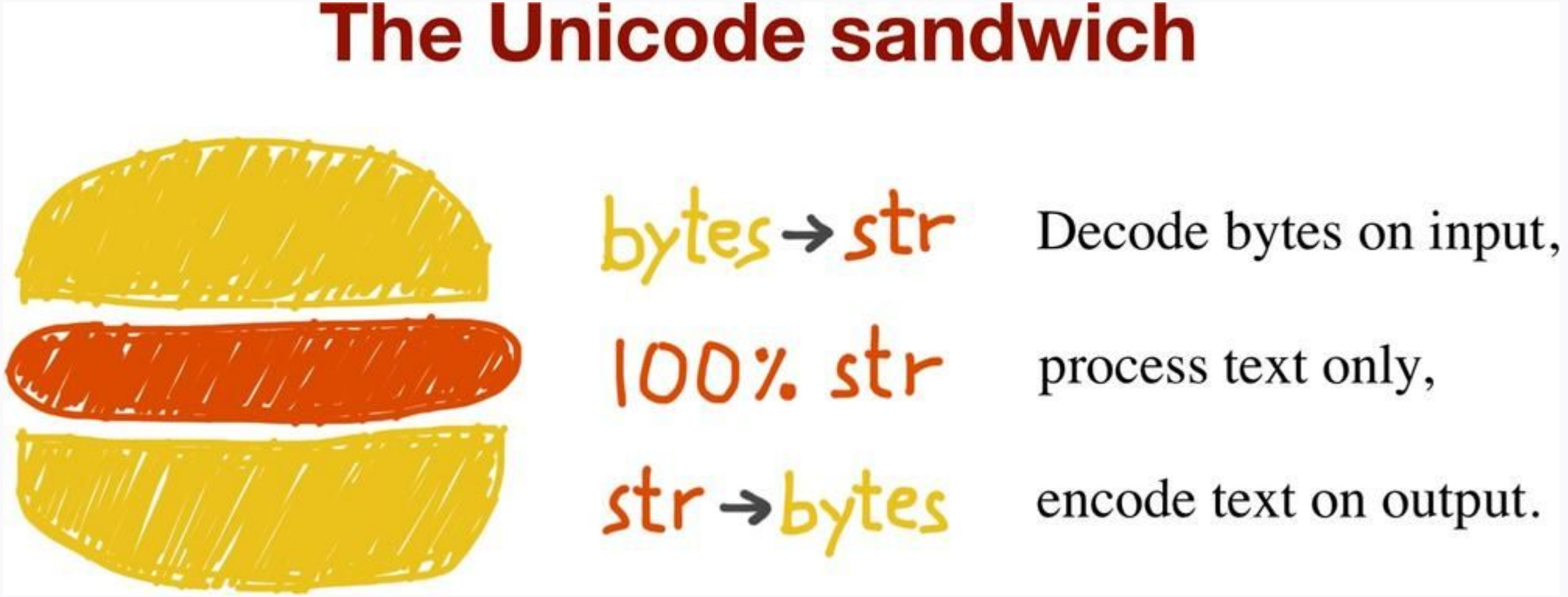
Повторим для закрепления

Конвертация из строки в байты и наоборот



Таким образом

Обработка данных в Python представляет собой Unicode-сэндвич



Практическое задание

Практическое задание

1. Каждое из слов «разработка», «сокет», «декоратор» представить в строковом формате и проверить тип и содержание соответствующих переменных. Затем с помощью онлайн-конвертера преобразовать строковые представление в формат Unicode и также проверить тип и содержимое переменных.
2. Каждое из слов «class», «function», «method» записать в байтовом типе без преобразования в последовательность кодов (не используя методы **encode** и **decode**) и определить тип, содержимое и длину соответствующих переменных.
3. Определить, какие из слов «attribute», «класс», «функция», «type» невозможно записать в байтовом типе.

Практическое задание (продолжение)

4. Преобразовать слова «разработка», «администрирование», «protocol», «standard» из строкового представления в байтовое и выполнить обратное преобразование (используя методы **encode** и **decode**).
5. Выполнить пинг веб-ресурсов yandex.ru, youtube.com и преобразовать результаты из байтовового в строковый тип на кириллице.
6. Создать текстовый файл **test_file.txt**, заполнить его тремя строками: «сетевое программирование», «сокет», «декоратор». Проверить кодировку файла по умолчанию. Принудительно открыть файл в формате Unicode и вывести его содержимое.

Дополнительные материалы

1. Таблица символов Юникода: http://foxtools.ru/Unicode.
2. Юникод-конвертер: https://www.branah.com/unicode-converter.
3. Устройство оперативной памяти компьютер: http://iguania.ru/article/ram.
4. Кодировка символов Юникод: https://www.ibm.com/support/knowledgecenter/ru/SSEPGG_9.1.0/com.ibm.db2.udb.admin.doc/doc/c0004816.htm.