

# JavaScript

---

**WEB  
COURSE  
ORT DNIPRO**

---

**ORTDNIPRO.ORG/WEB**

# 1. Интерактивное Программное Обеспечение

XEROX 6085 Workstation

User-Interface Design

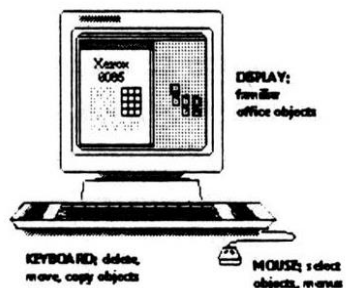
To make it easy to compose text and graphics, to do electronic filing, printing, and mailing all at the same workstation, requires a revolutionary user interface design.

**Bit-map display** - Each of the pixels on the 19" screen is mapped to a bit in memory; thus, arbitrarily complex images can be displayed. The 6085 displays all fonts and graphics as they will be printed. In addition, familiar office objects such as documents, folders, file drawers and in-baskets are portrayed as recognizable images.

**The mouse** - A unique pointing device that allows the user to quickly select any text, graphic or office object on the display.

See and Point

All functions are visible to the user on the keyboard or on the screen. The user does filing and retrieval by selecting them with the mouse and touching the MOVE, COPY, DELETE or PROPERTIES command keys. Text and graphics are edited with the same keys.



Shorter Production Times

Experience at Xerox with prototype work stations has shown shorter production times and thus lower costs, as a function of the percentage of use of the workstations. The following equation can be used to express this:

Year	Non 6085	6085
1978	85.2	15.8
1980	61.1	39.9
1982	45	55
1984	30	70
1986	10	90
1988	5	95

Table 1: Percentages of use of methods.

Activity under the old and the new



Figure 1: Data from Table 1 drive

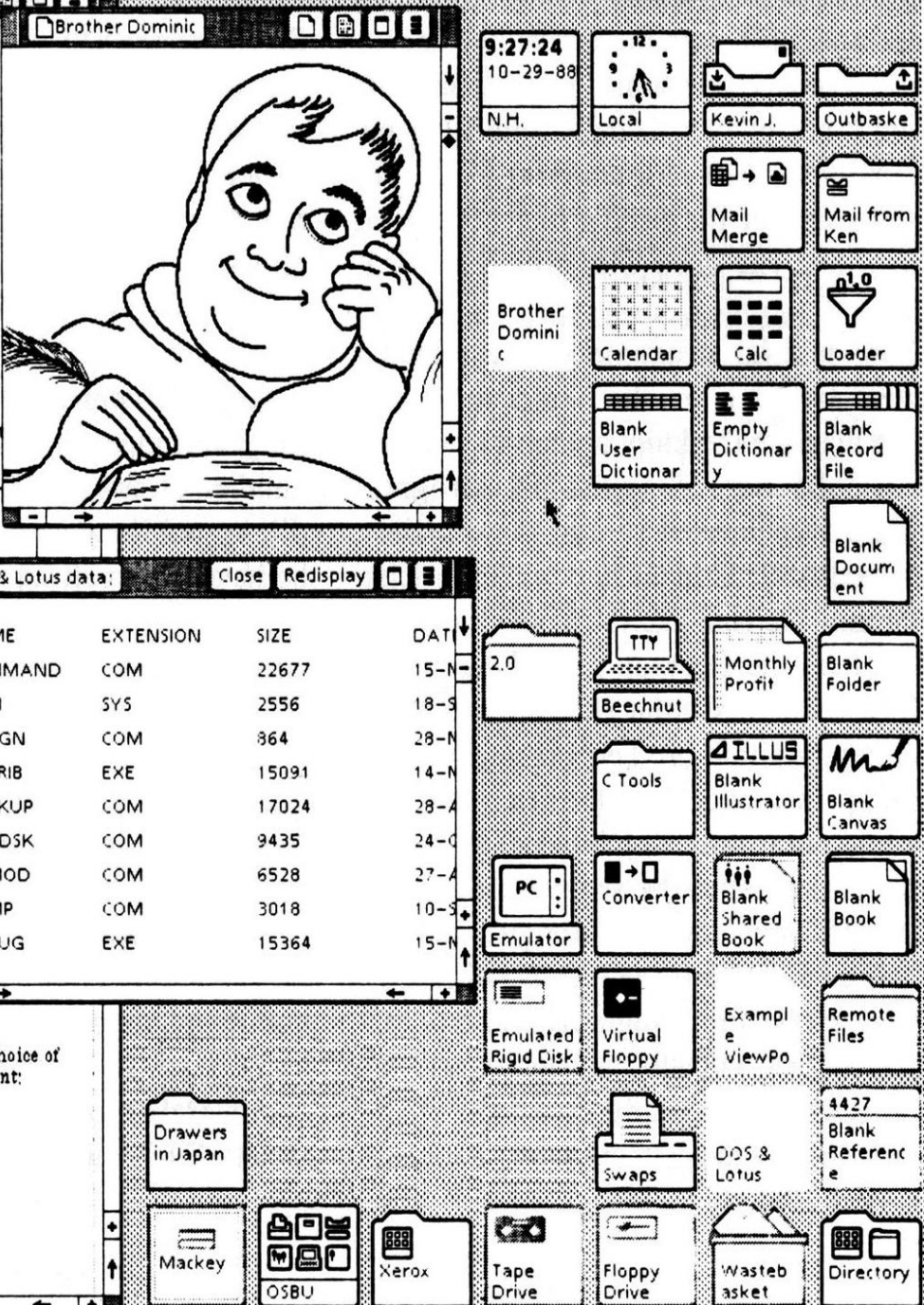
$$X(n) = \sum_{i=1}^n \frac{A + F \cdot P^i}{\text{denominator} + P^i}$$

Workstation usage percentages Table 1 and illustrated in Figure 6085 users are likely to do more composition and layout, control process including printing and di

Text and Graphics

To replace typesetting, the 6085 offers a choice of type fonts and sizes, from 6 point to 36 point:

Here is a sentence of 6-point text.  
Here is a sentence of 10-point text.  
18-point text.  
24-point text.  
36-point text.



Скевоморфизм - орнамент или элемент дизайна, который скопирован с формы другого объекта, но изготовлен из других материалов или иным методом. /Википедия.



Элементы интерфейса  
первых версий iOS.

**Скевоморфизм** -  
орнамент или  
элемент дизайна,  
который скопирован  
с формы другого  
объекта, но  
изготовлен из других  
материалов или  
иным методом.  
[/Википедия.](#)



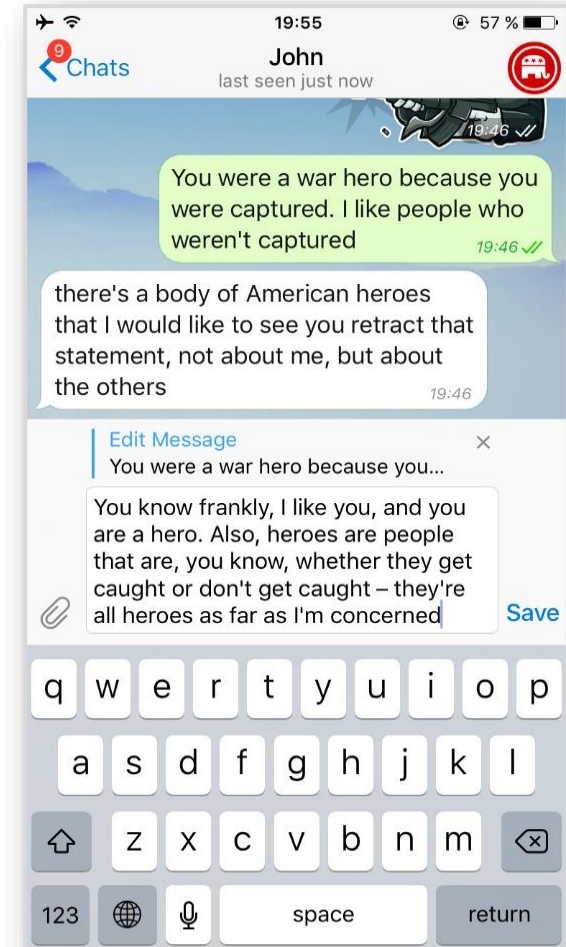
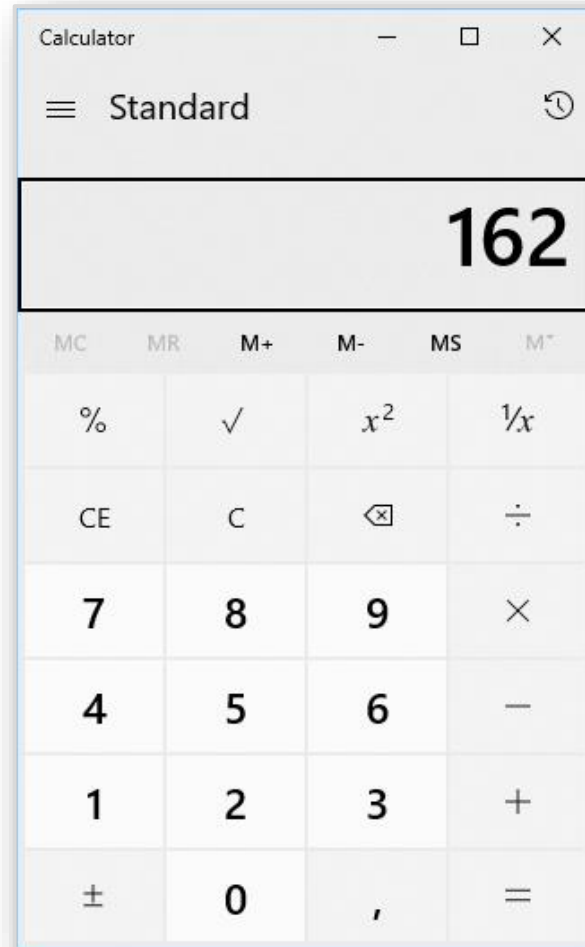
## 2. Событийная модель управления

# Интерактивное программное обеспечение и



**Событийная модель управления**

# Интерактивное программное обеспечение + **событийная модель управления**

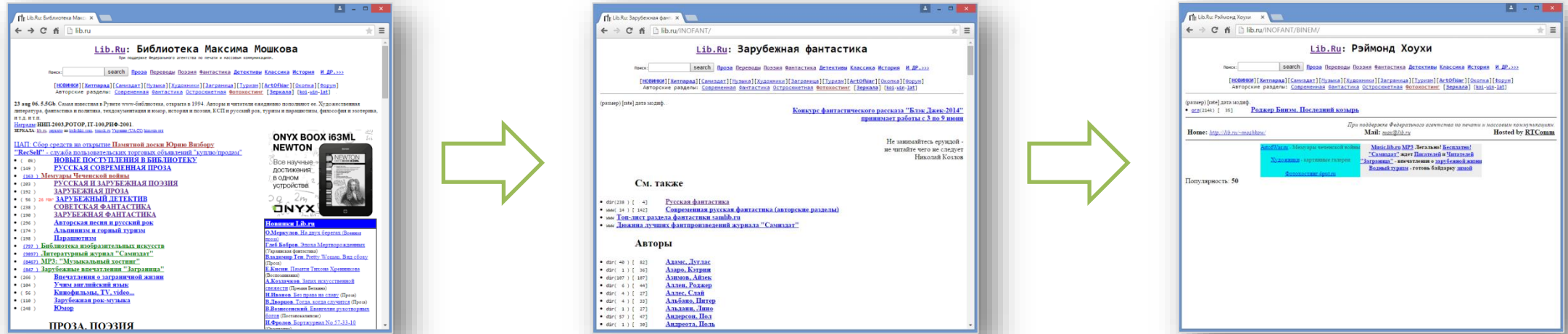


# 3. Интерактивное программное обеспечение ...в браузере...



# HTML статичен

Что неудивительно, ведь HTML (и CSS) не является языком программирования.



После того как страница загрузится в браузер она остаётся неизменной, информация на ней не изменяется. Чтобы получить другую информацию, нужно загрузить другую страницу. Однако пользователи (поработав с *настольным программным обеспечением*) привыкли к какой-никакой но **интерактивности**.

# HTML/CSS – декларативные языки

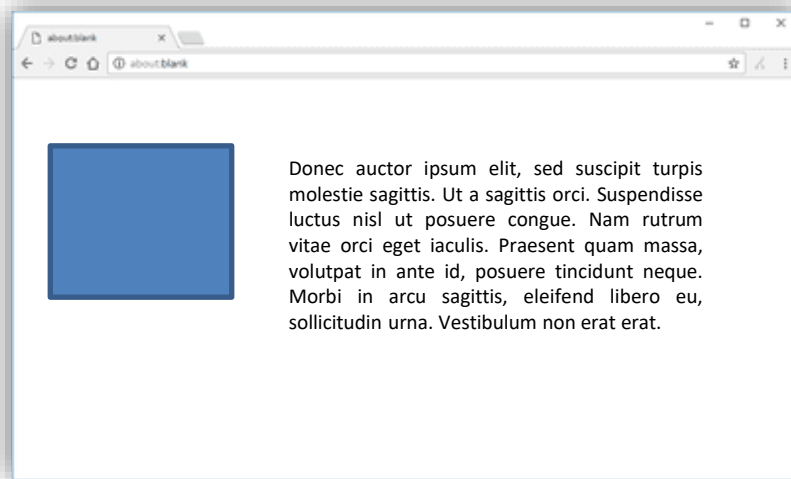
**Декларативные** языки при помощи директив позволяют указать какой результат мы хотим получить, но не путь его достижения (путь его достижения определяет компьютер). Побочный эффект: всё что не предусмотрено имеющимися директивами – реализовать не получится.

# JavaScript – императивный язык

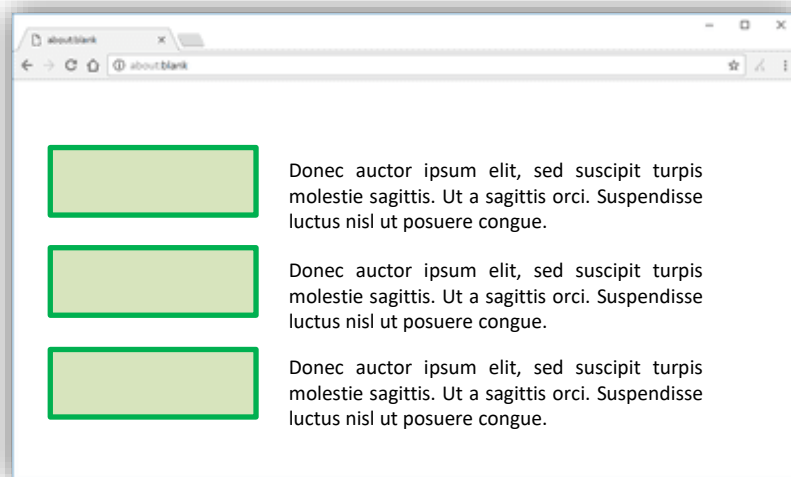
**Императивные** языки (как правило языки программирования относятся к этой категории) – языки состоящие из инструкций (малых действий, «шагов») выполняемых в определённой последовательности. Тем самым код на императивном языке описывает путь достижения желаемого результата.

**Отличительная черта императивных языков:** работа (обработка) данных, описание логики (ветвления) работы программы.

# Задачи JavaScript?



Изменение  
содержимого  
страницы (*в первую  
очередь **данных***).  
А за содержимое  
страницы отвечает  
**разметка...**



# Задачи JavaScript

1. Манипуляция элементами (тегами) HTML-страницы  
(когда страница уже в браузере посетителя);

*А если конкретнее, то: **изменять разметку документа**. Ведь браузер «нарисует» только то что описано в разметке.*

2. Делать что-то в ответ на действия пользователя  
(реагировать на действия пользователя).

# 4. На практике



Hello, JS!



Click on the photo to display the next one

Photo #1

## «Листалка» фотографий

Задача **JavaScript** — изменять разметку страницы, ссылка на изображение в теге `<img src='...'>` тоже относиться к разметке страницы...

Воспользуйтесь шаблоном в репозитории занятия [./src/js-demo-example/](#)

# 5. JavaScript – язык программирования

# HTML



# CSS



# JS



**Три базовые технологии Front End разработки**

# JavaScript – язык программирования

*его «составные части»*

**Переменные / Типы / Операции**

**Ветвления (условные операторы)**

**Циклы / Массивы (структуры данных)**

**Функции**

**Объекты**

# 6. Алгоритм

# Алгоритм

**Задача:** Скрипт должен рассчитывать сколько гривен в день приносит вкладчику депозит размещенный на полтора года по ставке в 20% годовых?

## Проблемы:

- ✓ *Дан недостаточный объём данных или часть данных задана неявно, нужно уточнять;*
- ✓ *Часть данных избыточна;*
- ✓ *Есть сторонние факторы, не известные заранее, но влияющие на результат.*



# Алгоритмы

**Задача:** Скрипт должен рассчитывать сколько гривен в день приносит вкладчику депозит размещенный на **полтора года** по ставке в **20%** годовых?

## Алгоритм:

1. Уточняем сумму депозита.

2. Рассчитываем сколько будет дохода за целый год:

$$\text{Доход} = \text{Сумма} * (20\% / 100);$$

3. Считаем доход за 1 день:

$$\text{Доход\_день} = \text{Доход} / \text{Количество\_дней\_в\_году};$$

4. Рассчитываем налоги:

$$\text{Сумма\_налога} = \text{Доход\_день} * ((18\% + 1,5\%) / 100);$$

5. Учитываем налог:

$$\text{Доход\_день\_после\_налога} = \text{Доход\_день} - \text{Сумма\_налога};$$

6. Выводим результаты.

**Будет полезным**

# Prometheus | Harvard CS50 | v. 2019



this is  
**CS50**  
harvard college - fall 2011



HARVARD

School of Engineering  
and Applied Sciences

**CS50**

Вводный курс по  
компьютерным  
наукам (*Computer  
Science*) и основам  
программирования  
от Гарвардского  
университета.

[https://courses.prometheus.org.ua/courses/course-v1:Prometheus+CS50+2019\\_T1/about](https://courses.prometheus.org.ua/courses/course-v1:Prometheus+CS50+2019_T1/about)

**На следующем занятии**

# JS: Переменные и операции

# Javascript



```
<script type="text/javascript">
  switch (new Date().getDay()) {
    case 6:
      text = "Friday";
      break;
    case 0:
      text = "Sunday";
      break;
    default:
      text = "Choose Your Day";
  }
</script>
```