Ростовский Государственный Строительный Университет

Языки Программирования Высокого Уровня

Презентацию подготовил Колыванов Андрей ученик 9 Б класса

Введение

Создатели языков по-разному толкуют понятие язык программирования. К наиболее распространенным утверждениям, признаваемым большинством разработчиков, относятся следующие:

Функция: язык программирования предназначен для написания компьютерных программ, которые применяются для передачи компьютеру инструкций по выполнению того или иного вычислительного процесса и организации управления отдельными устройствами.

Задача языка программирования отличается от естественных языков тем, что предназначен для передачи команд и данных от человека компьютеру, в то время, как естественные языки используются для общения людей между собой. В принципе, можно обобщить определение «языков программирования» — это способ передачи команд, приказов, четкого руководства к действию, тогда как человеческие языки служат также для обмена информацией.

***Исполнение*:** язык программирования может использовать специальные конструкции для определения и манипулирования структурами данных и управления процессом вычислений.

Введение

Список языков программирования:

Классы языков программирования:

1. Функциональные
2. Императивные
3. Стековые
4. Процедурные
5. Логического программирования
6. Декларативные
7. Динамические
8. Учебные
9. Прототипные
10. Объектно-ориентированные
11. Рефлексивные
12. Логического программирования
13. Параллельного программирования
14. Сценарные (скриптовые)
15. Эзотерические

1. Неклассифицированные языки
2. XML-подобные языки программирования
3. Структурные языки программирования
4. Процедурные языки программирования
5. Логические языки программирования
6. Программирование в ограничениях
7. Аспектно-ориентированные
8. Объектно-ориентированные языки программирования
8. Функциональные языки программирования
9. Языки программирования для промышленной автоматизации
10. Эзотерические языки программирования
11. Стековые языки
12. Параллельные языки программирования

Введение

Неклассифицированные языки:

ABAP/4
Awk
BCPL
FoxPro
Tcl/Tk
ML
MQL4
PostScript
PL/SQL
Clarion
Clean
Clipper
Curry
Gentee
GPSS
DataFlex
Erlang

JOVIAL
Jython
Mercury
Linda
Occam
ПЛ/1
PL/M
Pixilang
Планкалкюль
Scala
xBase
Progress 4gl
X++
Sieve
PureBasic — компилируемая модификация Basic
Ассемблеры
Visual DataFlex

Робик
Рапира
УА (Упрощённый Алгол)
Nemerle
ДРАКОН
ORACLE

Введение

XML-подобные языки программирования

ApplicationXML

Структурные языки программирования

Алгол
Алгол 68
Basic
QBASIC
Фортран
REXX
sh
Фокал

Введение

Процедурные языки программирования

Алгоритмический язык

Би (язык программирования)

Си

КОБОЛ

Limbo

Lua

Maple

MATLAB

Модула-2

Паскаль

Логические языки программирования

Prolog

Введение

Функциональные языки программирования

Лисп
Cat (Stack-oriented programming language)
Лого
Dylan
Haskell
OCaml
Scheme
РЕФАЛ
АПЛ
J
Hope

Стековые языки

PostScript
Forth

Параллельные языки программирования

MC#

Языки программирования для промышленной автоматизации

(стандарта IEC61131-3)
FBD
IL
ST или SCL
Sequential Function Chart
Ladder Diagram
SPCLK

Эзотерические языки программирования

Byter
Brainfuck
Befunge
INTERCAL
FALSE
Whitespace
Piet

Стандартизация языков программирования

Язык программирования может быть представлен в виде набора спецификаций, определяющих его синтаксис и семантику.

Для многих широко распространённых языков программирования созданы международные стандарты. Специальные организации проводят регулярное обновление и публикацию спецификаций и формальных определений соответствующего языка. В рамках таких комитетов продолжается разработка и модернизация языков программирования и решаются вопросы о расширении или поддержке уже существующих и новых языковых конструкций.

Типы данных

Во внутреннем представлении обычно данные в современных цифровых компьютерах сохраняются в бинарном виде (в двоичном виде). Данные, которые представляют информацию из реального мира (имена, банковские счета, измерения и др.) высокоуровневые концепции.

Особая система, по которой данные организуются в программе, эта система типов языка программирования; разработка и изучение систем типов известна под названием теория типов. Языки могут быть классифицированы как системы со статической типизацией и языки с динамической типизацией.

Статически-типизированные языки могут быть в дальнейшем подразделены на языки с обязательной декларацией, где каждая переменная и объявление функции имеет обязательное объявление типа, и языки с выводимыми типами. Иногда динамически-типизированные языки называются латентно типизированными

Структуры данных

Системы типов в языках высокого уровня позволяют определять сложные, составные типы, так называемые структуры данных. Как правило, структурные типы данных образуются как декартово произведение базовых (атомарных) типов и ранее определённых составных типов.

Основные структуры данных (списки, очереди, хэш-таблицы, двоичные деревья и пары) часто представлены особыми синтаксическими конструкциями в языках высокого уровня. Такие данные структурируются автоматически.

Семантика языков программирования

Существует несколько подходов к определению семантики языков программирования.

Наиболее широко распространены разновидности следующих: операционного (или так называемого математического), и деривационного (или аксиоматического).

При описании семантики в рамках операционного подхода обычно исполнение конструкций языка программирования интерпретируется с помощью некоторой воображаемой (абстрактной) ЭВМ.

Деривационная семантика описывает последствия выполнения конструкций языка с помощью языка логики и задания пред- и постусловий. Денотационная семантика оперирует понятиями, типичными для математики - множества, соответствия и др.

Парадигмы программирования

Язык программирования строится в соответствии с той или иной базовой моделью вычислений.

Несмотря на то, что большинство языков ориентировано на так называемую императивную модель вычислений, задаваемую так называемой фон-неймановской архитектурой ЭВМ, существуют и другие подходы. Прежде всего следует упомянуть языки со стековой вычислительной моделью (Forth, Factor, Postscript и др.), а также функциональное (Лисп, Haskell, ML и др.) и логическое программирование

В настоящее время также активно развиваются проблемно-ориентированные, декларативные и визуальные языки программирования.

Компилируемые и интерпретируемые языки

Как видно из названия, ***языки программирования могут быть разделены на компилируемые и интерпретируемые.***

Программа на *компилируемом* языке при помощи специальной программы компилятора преобразуется (компилируется) в набор инструкций для данного типа процессора (машинный код) и далее записывается в исполняемый файл, который может быть запущен на выполнение как отдельная программа. Другими словами, компилятор переводит программу с языка высокого уровня на низкоуровневый язык, понятный процессору. Если программа написана на *интерпретируемом* языке, то интерпретатор непосредственно выполняет (интерпретирует) ее текст без предварительного перевода. При этом программа остается на исходном языке и не может быть запущена без интерпретатора. Можно сказать, что *Однако интерпретируемые программы работают заметно медленнее, чем компилируемые, кроме того, они не могут выполняться без дополнительной программы-интерпретатора*.

Языки высокого уровня

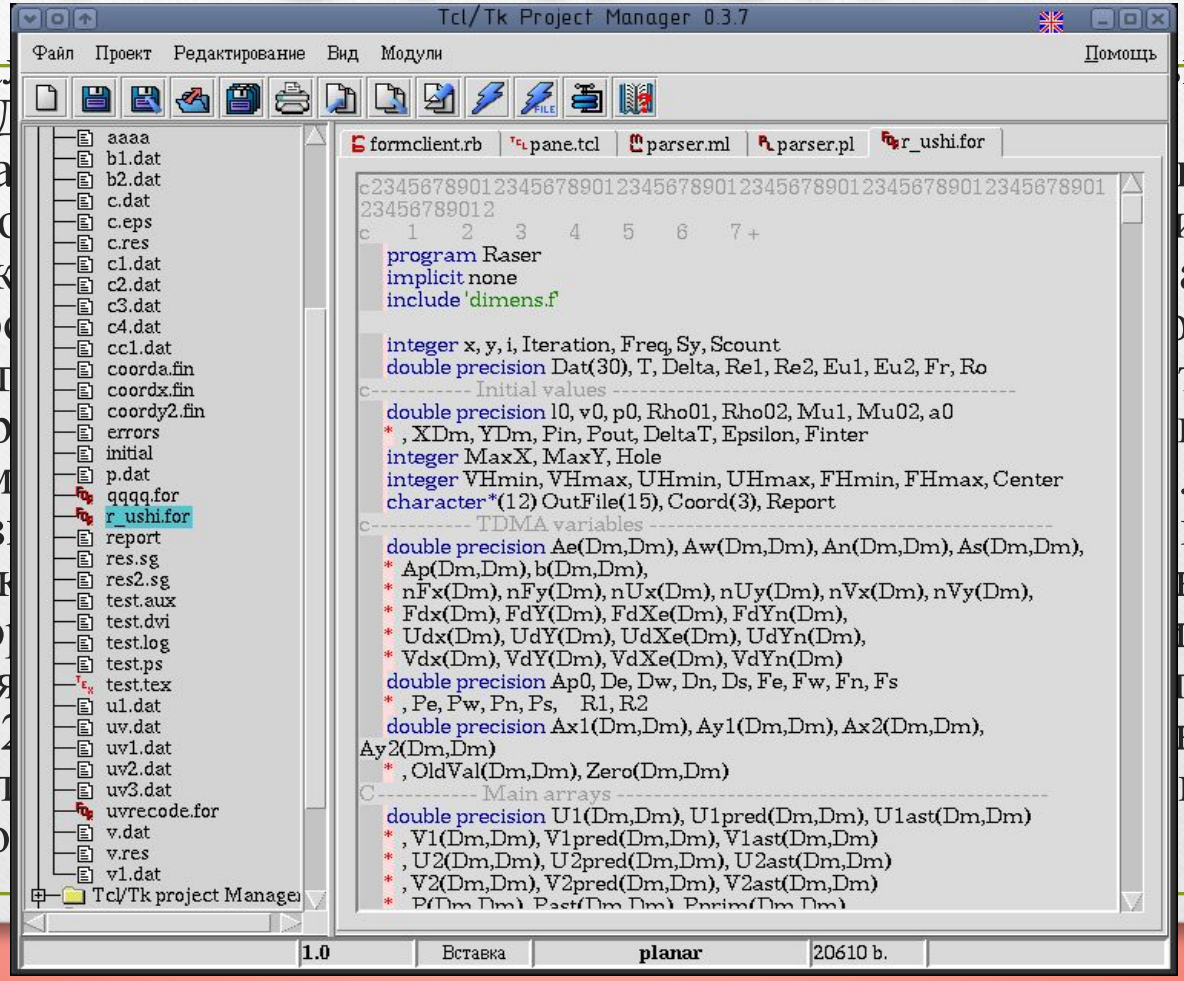
Высокоуровневый язык программирования - язык программирования, разработанный для быстроты и удобства использования программистом. Основная черта высокоуровневых языков - это абстракция, то есть введение смысловых конструкций, кратко описывающих такие структуры данных и операции над ними, описания которых на машинном коде (или другом низкоуровневом языке программирования) очень длинны и сложны для понимания.

Так, высокоуровневые языки стремятся не только облегчить решение сложных программных задач, но и упростить сортирование программного обеспечения. Использование разнообразных трансляторов и интерпретаторов обеспечивает связь программ, написанных при помощи языков высокого уровня, с различными операционными системами и оборудованием, в то время как их исходный код остаётся, в идеале, неизменным.

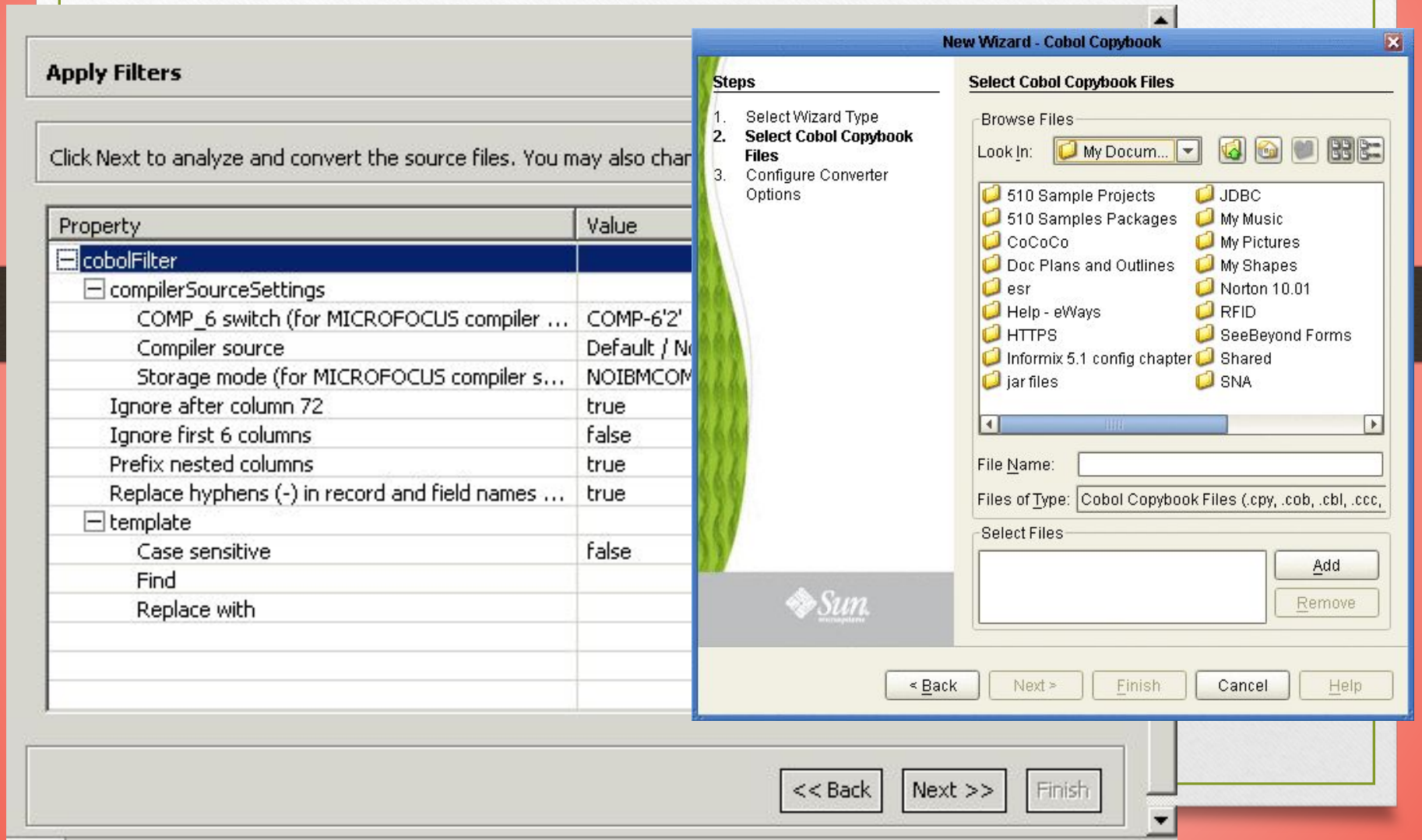
Языки высокого уровня

Fortran

Первый реа… …ня
созданный Д…
разрабатыва… …ражали
серьезное со… …ительного
языка высок… …аботке
компилятор… …о хода.
Хотя в Форт… …тий
программир… …но в
жертву возм… …. Однако
для этого яз… …начиная
от статическ… …никами,
поэтому Фо… …их
организация… …том
Фортрана F2… …ная
версия Форт… …ых супер
компьютеро…



Cobol

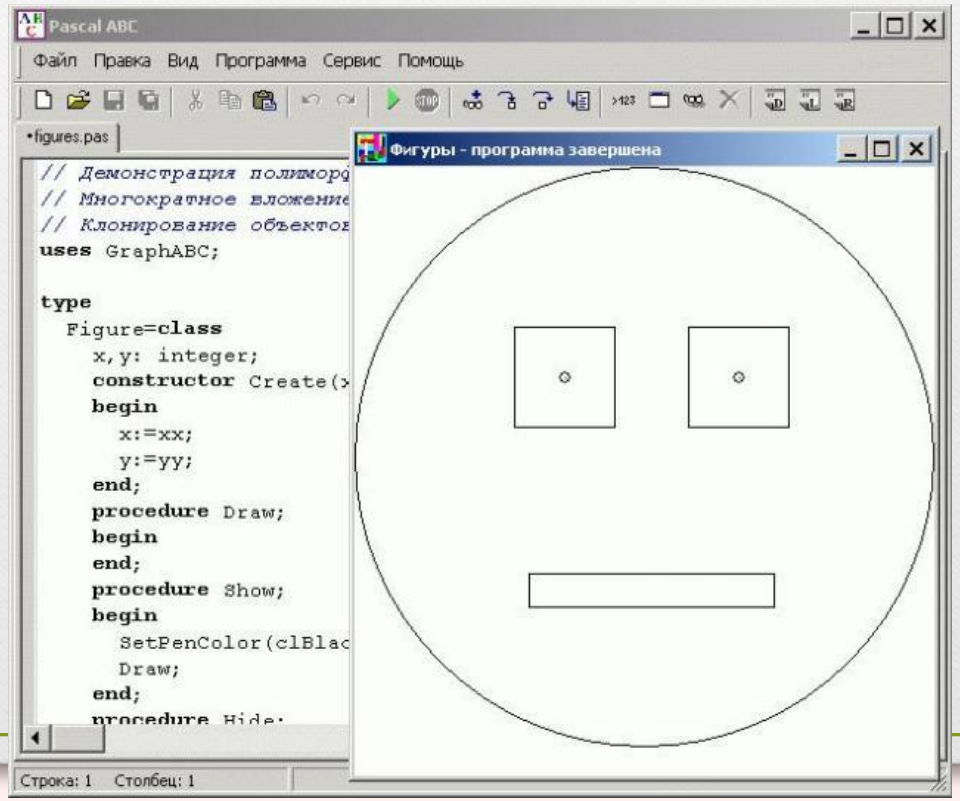

Apply Filters
Click Next to analyze and convert the source files. You may also char
Property
Value
cobolFilter
compilerSourceSettings
COMP_6 switch (for MICROFOCUS compiler ...
COMP-6'2'
Compiler source
Default / N
Storage mode (for MICROFOCUS compiler s...
NOIBMCOM
Ignore after column 72
true
Ignore first 6 columns
false
Prefix nested columns
true
Replace hyphens (-) in record and field names ...
true
template
Case sensitive
false
Find
Replace with
<< Back
Next >>
Finish
Main
Source
New Wizard - Cobol Copybook
Steps
1. Select Wizard Type
2. Select Cobol Copybook Files
3. Configure Converter Options
Select Cobol Copybook Files
Browse Files
Look In:
My Docum...
510 Sample Projects
510 Samples Packages
CoCoCo
Doc Plans and Outlines
esr
Help - eWays
HTTPS
Informix 5.1 config chapter
jar files
JDBC
My Music
My Pictures
My Shapes
Norton 10.01
RFID
SeeBeyond Forms
Shared
SNA
File Name:
Files of Type:
Cobol Copybook Files (.cpy, .cob, .cbl, .ccc,
Select Files
Add
Remove
< Back
Next >
Finish
Cancel
Help

Algol

Алгол (англ. Algol от англ. algorithmic - алгоритмический и англ. language - язык) - название ряда языков программирования, применяемых при составлении программ для решения научно-технических задач на ЭВМ. Разработан комитетом по языку высокого уровня IFIP в 1958-1960 гг. (Алгол-58, Алгол-60); усовершенствован в 1964-1968 гг. (Алгол 68). Алгол относится к языкам высокого уровня и позволяет легко переводить алгебраические формулы в программные команды. Алгол был популярен в Европе, в том числе в СССР, в то время как сравнимый с ним язык Фортран был распространен в США и Канаде. Оказал заметное влияние на все разработанные позднее императивные языки программирования - в частности, на язык Pascal.

Pascal

Один из наиболее известных языков программирования, широко применяется в промышленном программировании, обучении программированию в высшей школе, является базой для большого числа других языков. Был создан Никлаусом Виртом в 1970, после его участия в работе комитета разработки стандарта языка Алгол-68.



Pascal

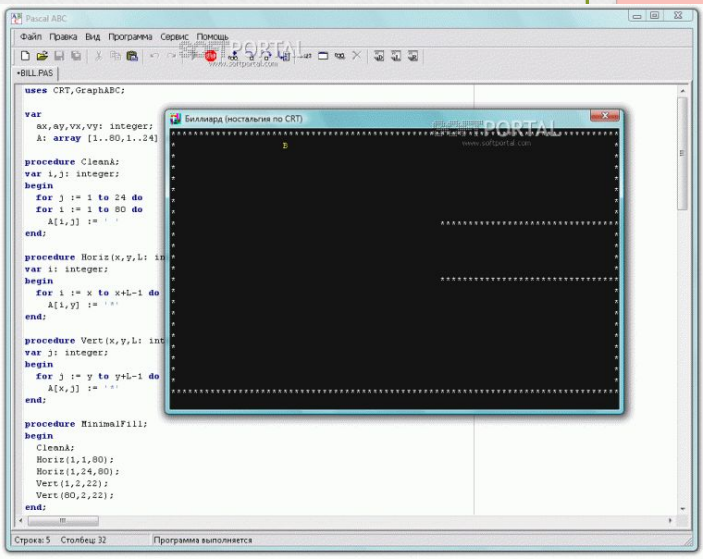
Особенностями языка являются строгая типизация и наличие средств структурного (процедурного) программирования. Паскаль был одним из первых таких языков. По мнению Н. Вирта, язык должен способствовать дисциплинированию программирования, поэтому, наряду со строгой типизацией, в Паскале сведены к минимуму возможные синтаксические неоднозначности, а сам синтаксис автор постарался сделать интуитивно понятным даже при первом знакомстве с языком.

Паскаль был создан как язык для обучения процедурному программированию (хотя, по словам Вирта, язык нельзя считать только учебным, поскольку язык, непригодный для написания реальных программ, для обучения использоваться не должен). Название языку дано в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля. Один из первых языков, для которых была создана реализация «на самом себе» — компилятор Паскаля был написан на самом Паскале. В начале 1970-х годов для переноса Паскаль-систем на различные аппаратные платформы была создана система Pascal-P, в которой был единый компилятор Паскаля в промежуточный язык (P-код) и для каждой платформы создавался быстрый интерпретатор P-кода. Заимствование этой системы привело к созданию системы UCSD Pascal в Университете Сан-Диего (Калифорния, США), намного позже её идеи были заимствованы создателями языка Java (байт-код, компиляция в байт-код, интерпретатор байт-кода).

Pascal

Тем не менее, первоначально язык имел ряд ограничений: невозможность передачи функциям массивов переменной длины, отсутствие нормальных средств работы с динамической памятью, ограниченная библиотека ввода-вывода, отсутствие средств для подключения функций написанных на других языках, отсутствие средств раздельной компиляции и т. п. Наиболее бросающийся в глаза недостаток синтаксиса — некритически заимствованная из Алгола структура управляющих конструкций (операторов if и циклов), требующая, как правило, постоянного использования составных операторов «begin — end». Полный разбор недостатков языка Паскаль был выполнен Брайаном Керниганом в статье «Почему Паскаль не является моим любимым языком программирования» (интересно, что эта статья вышла в начале 1980-х, когда уже существовал язык Модула-2, потомок Паскаля, избавленный от большинства его пороков, а также более развитые диалекты Паскаля). Некоторые недостатки Паскаля были исправлены в ISO-стандарте 1982 года, в частности, в языке появились открытые массивы, давшие возможность использовать одни и те же процедуры для обработки одномерных массивов различных размеров.

Необходимо заметить, что многие недостатки языка не проявляются или даже становятся достоинствами при обучении программированию. Кроме того, по сравнению с основным языком программирования в академической среде 70-х (которым был Фортран, обладавший гораздо более существенными недостатками), Паскаль представлял собой значительный шаг вперёд. В начале 1980-х годов в СССР для обучения школьников основам информатики и вычислительной техники академик А. П. Ершов разработал алголо-паскалеподобный «алгоритмический язык».

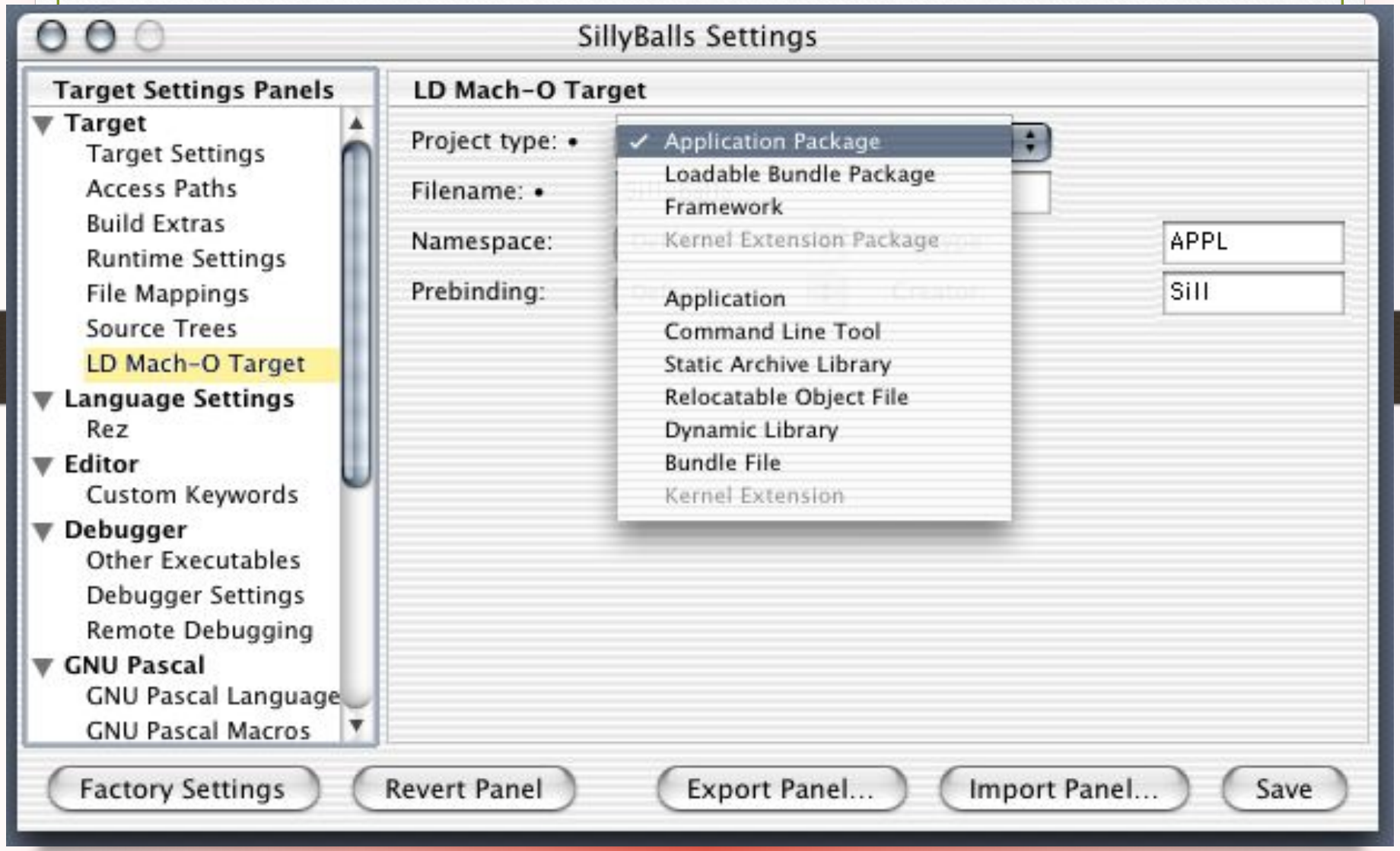


Pascal

Никлаус Вирт понимал недостатки созданного им языка, но, следуя традициям академической среды и собственным принципам, согласно которым «неподходящий инструмент надо не исправлять, а заменять», не стал его развивать дальше, а разработал новые языки семейства: Модула-2 и Оберон. В противоположность этому промышленные традиции и достоинства языка побудили многие коммерческие и некоммерческие организации продолжать разрабатывать и развивать системы программирования именно на основе языка Паскаль, подвергая язык произвольному расширению, добавляя в него, часто совершенно механически, новые средства и синтаксические конструкции.

Наиболее известной реализацией Паскаля являлась система Turbo Pascal (выросшая затем в Borland Pascal для DOS/Windows и далее в Delphi) фирмы Borland, в которой использовались значительные расширения языка. Благодаря появлению развитых диалектов, язык стал богаче, но в отсутствие отраслевой стандартизации, потерял переносимость и общность (только в 1995 году появилась совместимая с Borland Pascal версия среды разработки Virtual Pascal для OS/2 и впоследствие Linux, в 1998 году Kylix — Delphi для Linux, в настоящее время оба этих проекта фактически заморожены).

Pascal



GNU Pascal

BASIC

BASIC — сокращение от англ. Beginner's All-purpose Symbolic Instruction Code — универсальный код символических инструкций для начинающих; англ. basic — основной, базовый) — семейство высокоуровневых языков программирования.

Был разработан в 1963 профессорами Дартмутского колледжа Томасом Куртцом (Thomas E. Kurtz, 1928-) и Джоном Кемени (John G. Kemeny, 1926—1993). Язык предназначался для обучения программированию и получил широкое распространение в виде различных диалектов, прежде всего, как язык для домашних микрокомпьютеров.

Бейсик был спроектирован так, чтобы студенты могли писать программы, используя терминалы с разделением времени. Он создавался как решение для проблем, связанных со сложностью более старых языков. Он предназначался для более «простых» пользователей, не столько заинтересованных в скорости программ, сколько просто в возможности использовать компьютер для решения своих задач.

BASIC

При проектировании языка использовались следующие восемь принципов, новый язык должен:

1. быть простым в использовании для начинающих
2. быть языком программирования общего назначения
3. предоставлять возможность расширения функциональности, доступную опытным программистам
4. быть интерактивным
5. предоставлять ясные сообщения об ошибках
6. быстро работать на небольших программах
7. не требовать понимания работы аппаратного обеспечения
8. защищать пользователя от операционной системы

BASIC

Язык был основан частично на Фортран II и частично на Алгол-60, с добавлениями, делающими его удобным для работы в режиме разделения времени и, позднее, обработки текста и матричной арифметики. Первоначально Бейсик был реализован на мейнфрейме GE-265 с поддержкой множества терминалов. Вопреки распространённому убеждению, в момент своего появления это был компилируемый язык.

Взрывной рост

Несмотря на то, что язык уже использовался на нескольких миникомпьютерах, его настоящее распространение началось с его появления на микрокомпьютере Altair 8800. Многие языки программирования были слишком большими чтобы поместиться в небольшую память, которую пользователи таких машин могли себе позволить. Для машин с таким медленным носителем как бумажная лента (позднее — аудиокассета) и без подходящего текстового редактора такой небольшой язык как Бейсик был отличной находкой.

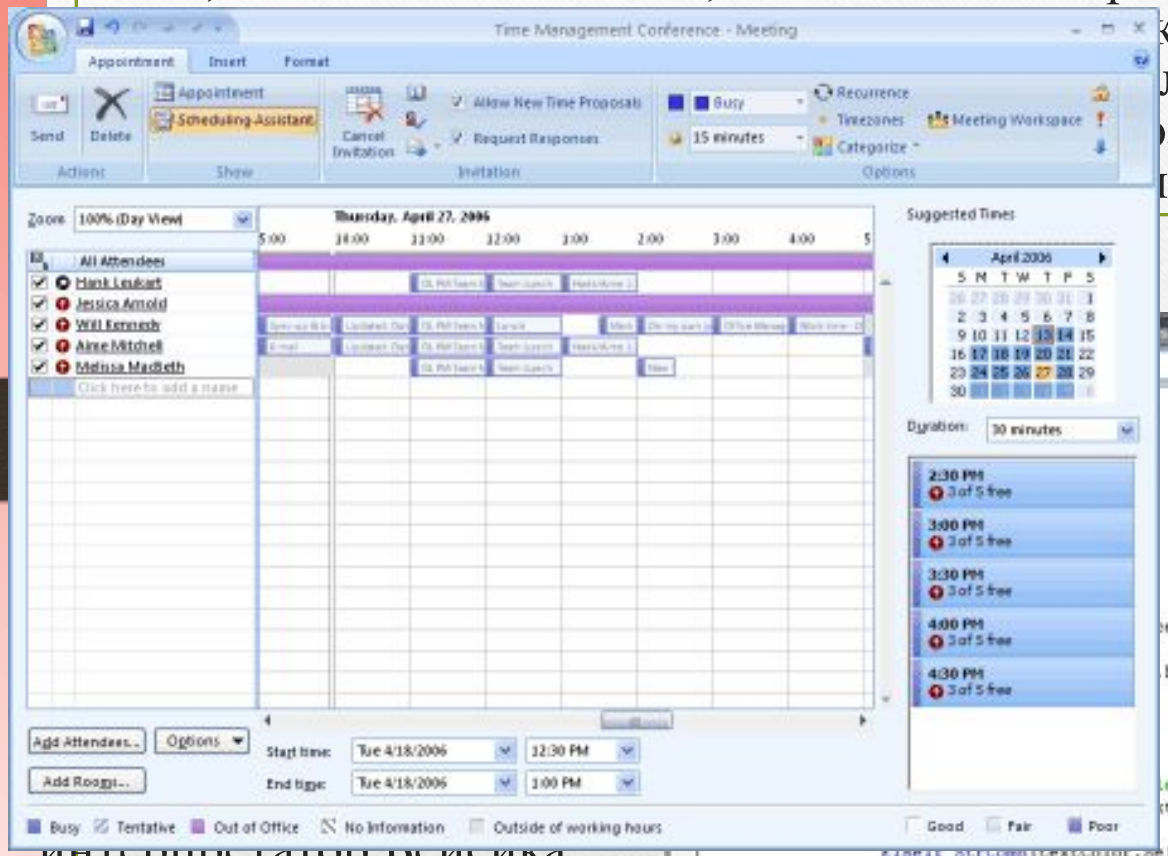
BASIC

Зрелость

В этот период было создано несколько новых версий Бейсика. Майкрософт продавала несколько версий Бейсик для MS-DOS/PC-DOS, включая BASICA, GWBASIC (модификация BASICA, не требующая «прошивки» от IBM) и Quick BASIC (QBASIC). Borland, известная своим Turbo Pascal, в 1985 выпустила Turbo BASIC 1.0 (его наследники впоследствии продавались другой компанией под именем PowerBASIC). На домашних компьютерах появились различные расширения Бейсика, обычно включающие средства для работы с графикой, звуком, выполнением DOS-команд, а также средства структурного программирования. Некоторые другие языки использовали хорошо известный синтаксис Бейсика в качестве основы, на которой строилась совершенно иная система (см. например, GRASS).

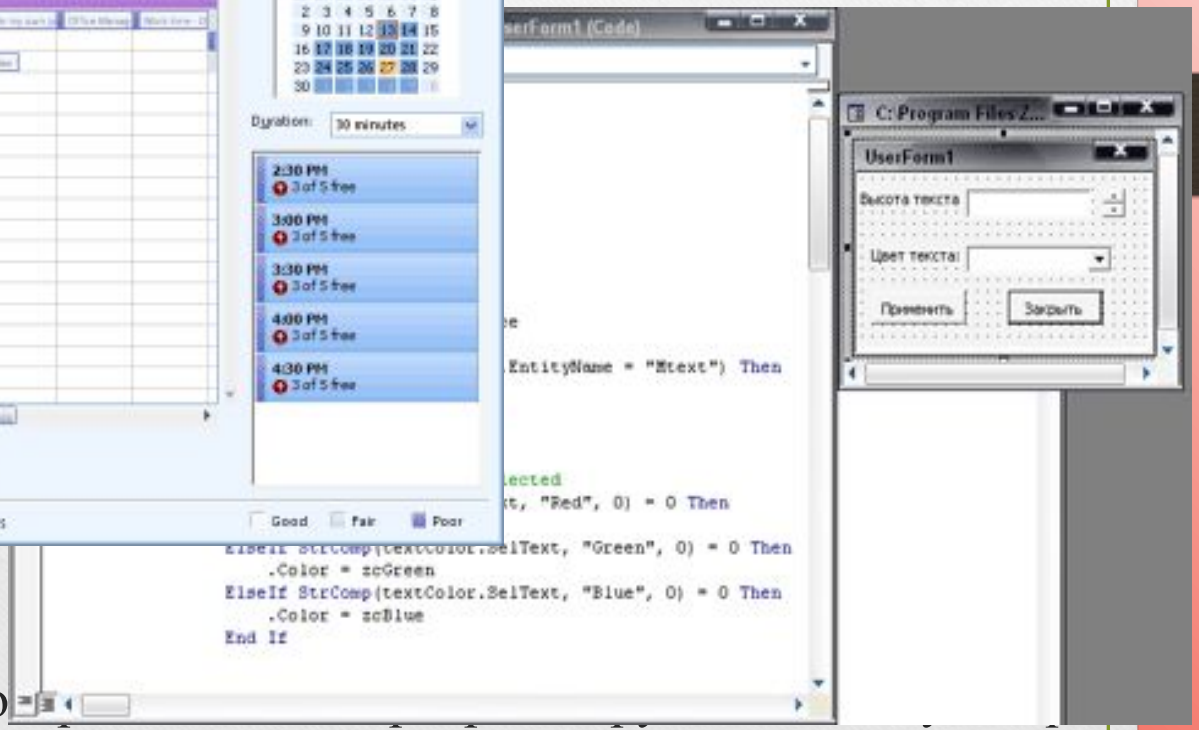
BASIC

Однако, начиная с конца 80-х, новые компьютеры стали намного более
как графический интерфейс
ль удобным для
зиции, несмотря на то, что
ось и продавалось.



интерпретатор Бейсика.

Microsoft Outlook



Бейсик используется в неко
Например, он встроен в отечественный калькулятор «Электроника МК-85».

VBA

Prolog

Пролог (Prolog) — язык логического программирования, основанный на логике дизъюнктов Хорна, представляющей собой подмножество логики предикатов первого порядка [illegible] за счет ООП-ар [illegible] ые реализа [illegible] ак ISO/IEC

Базовым [illegible] и данных [illegible] и записям [illegible] особы их обра [illegible] темы Пролога [illegible] и считают [illegible]

Пролог [illegible] решения на обычных языках программирования (типа C++, Java) в сочетании с базами данных оказываются более технологичными, чем аналогичные решения на Прологе. Негибкость заключается в трудности изучения языка, более высоких требований к квалификации программиста на Прологе, трудности отладки программы, неразвитости технологии программирования, плохой контролируемости промежуточных результатов.

Си++

Си++ представляет собой интересный эксперимент по адаптации возможностей объектной технологии к традиционному языку программирования. Бьерн Страуструп вполне достоин аплодисментов за то, что ему в голову пришла мысль слить обе технологии воедино. В то же время в Си++ сохранились проблемы старого поколения средств программного производства. Язык Си++ обладает тем преимуществом перед Си, что поддерживает некоторые аспекты объектной технологии, которые могут быть использованы для ограниченного проведения анализа требований и проектирования. Однако процессы анализа, проектирования и реализации проекта все еще в значительной степени остаются внешними по отношению к Си++. Таким образом, в Си++ не реализованы важные преимущества объектной технологии, которые прямо бы привели к экономичному производству программной продукции.

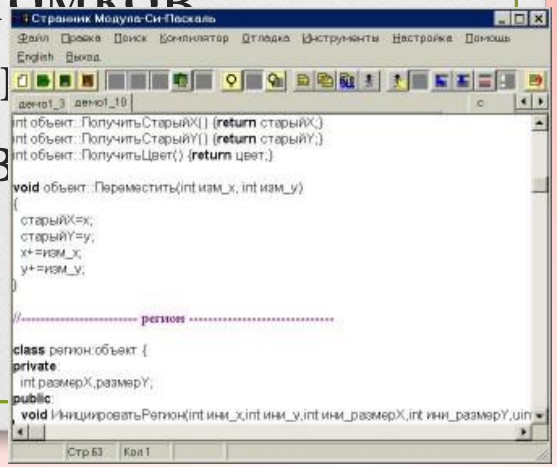
Полиморфизм - основополагающая концепция объектно-ориентированного программирования. В языке Си++ ключевое слово virtual предоставляет функции возможность стать полиморфической, если она будет переписана (переопределена) в одном классе-потомке или более. Однако слово virtual отнюдь не является необходимым, так как любая функция, переопределенная (overriden) в классе-потомке, может быть полиморфической. Компилятору только требуется генерировать коммутирующий код для истинно полиморфических процедур.

Если автор родительского класса в языке Си++ не предвидит, что класс-потомок захочет переопределить функцию, то он не сможет сделать ее и полиморфической. В этом заключается наиболее серьезный порок Си++, поскольку снижается гибкость программных компонентов, а следовательно, и способность создавать адаптируемые и расширяемые библиотеки.

Си++ также позволяет функциям быть перегруженными (overloaded); в такой ситуации вызов нужной функции зависит от аргументов. Различие между перегруженными и полиморфическими (переопределенными) функциями состоит в том, что в перегруженных функциях нужная определяется при компиляции, а в случае полиморфических определяется при выполнении.

Си++

Виртуальные функции предоставляют один из возможных путей реализации полиморфизма. Разработчик языка может сделать выбор в пользу определения полиморфизма либо в родительском, либо в наследующем классе. Так что же из них имеет смысл выбрать разработчику языка? Здесь можно выделить несколько вариантов для родительских классов и классов-потомков, которые не станут взаимоисключающ достаточно легко найти себе место в объектно-ориентированном языке.



SQL

SQL (англ. Structured Query Language - язык структурированных запросов) -универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Вопреки существующим заблуждениям, SQL является информационно-логическим языком, а не языком программирования.

SQL основывается на реляционной алгебре.

Целью разработки было создание простого непроцедурного языка, которым мог воспользоваться любой пользователь, даже не имеющий навыков программирования. Собственно разработкой языка запросов занимались Дональд Чэмбэрлин (Donald D. Chamberlin) и Рэй Бойс (Ray Boyce). Пэт Селинджер (Pat Selinger) занималась разработкой стоимостного оптимизатора (англ. cost-based optimizer), Рэймонд Лори (Raymond Lorie) занимался компилятором запросов.

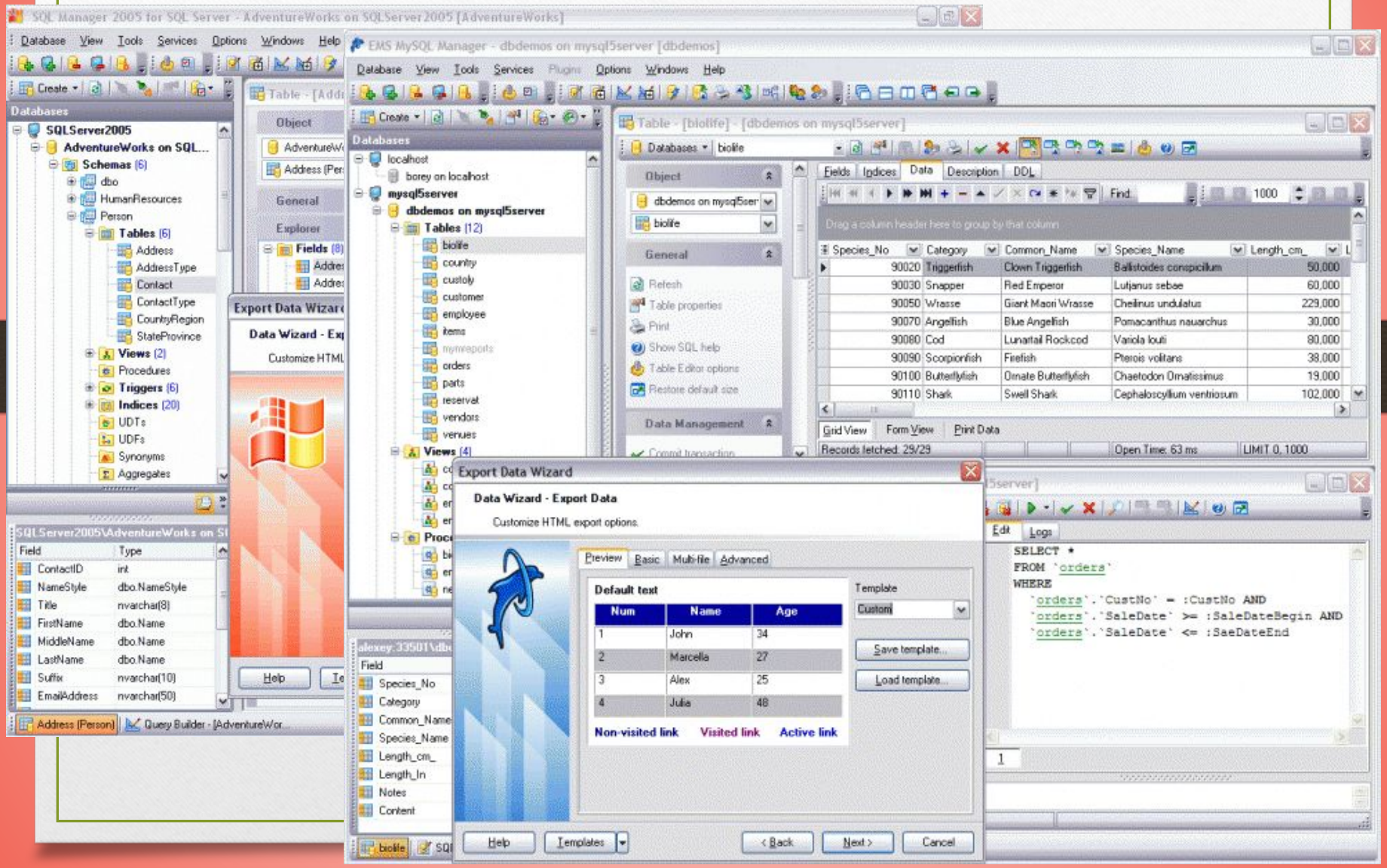
Как и с многими стандартами, имеющими место в IT-индустрии, с языком SQL возникла проблема, что в прошлом многие производители использующего SQL ПО решили, что функционал в текущей (на тот момент времени) версии стандарта недостаточен (что, в принципе, для ранних версий SQL было в некотором роде справедливо) и его желательно расширить. Что и приводит в данный момент к тому, что у разных производителей СУБД в ходу разные диалекты SQL, в общем случае между собой несовместимые.

До 1996 года вопросами соответствия коммерческих реализаций SQL стандарту занимался в основном институт NIST, который и устанавливал уровень соответствия стандарту. Но позднее подразделение, занимавшееся СУБД, было расформировано, и на текущий момент все усилия по проверке СУБД на соответствие стандарту ложатся на её производителя.

Впервые понятие «уровня соответствия» было предложено в стандарте SQL-92. А именно, ANSI и NIST определяли четыре уровня соответствия реализации этому стандарту:

1. Entry (русск. базовый)
2. Transitional (русск. переходный) — проверку на соответствие этому уровню проводил только институт NIST
3. Intermediate (русск. промежуточный)
4. Full (русск. полный)

SQL


SQL Manager 2005 for SQL Server - AdventureWorks on SQLServer2005 [AdventureWorks]
EMS MySQL Manager - dbdemos on mysql5server [dbdemos]
Export Data Wizard
Data Wizard - Export Data
Customize HTML export options.
Preview
Basic
Multi-file
Advanced
Default text
Num
Name
Age
1
John
34
2
Marcella
27
3
Alex
25
4
Julia
48
Non-visited link
Visited link
Active link
Template
Custom
Save template...
Load template...
Help
Templates
< Back
Next >
Cancel
SELECT *
FROM `orders`
WHERE
`orders`.`CustNo` = :CustNo AND
`orders`.`SaleDate` >= :SaleDateBegin AND
`orders`.`SaleDate` <= :SaeDateEnd

Со времени создания первых программируемых машин человечество придумало уже более восьми с половиной тысяч языков программирования. Каждый год их число пополняется новыми. Некоторыми языками умеет пользоваться только небольшое число их собственных разработчиков, другие становятся известны миллионам людей. Профессиональные программисты иногда применяют в своей работе более десятка разнообразных языков программирования.

Конец