**LabView**

**LabVIEW** — это один из основных продуктов компании [National Instruments](http://www.ni.com/). Прежде всего надо отметить, что LabVIEW — это аббревиатура, которая расшифровывается как **Lab**oratory **V**irtual **I**nstrumentation **E**ngineering **W**orkbench. Уже в названии прослеживается ориентация на лабораторные исследования, измерения и сбор данных. Действительно, построить SCADA — систему в LabVIEW несколько проще чем при использовании «традиционных» средств разработки. Рамки ее применения несколько шире, чем мы думаем. Это принципиально иной язык программирования, или если хотите целая «философия» программирования. Функциональный язык, заставляющий несколько иначе мыслить и порой предоставляющий совершенно фантастические возможности для разработчика. Является ли LabVIEW языком программирования вообще? Это спорный вопрос — здесь нет стандарта, как, например ANSI C. В узких кругах разработчиков мы говорим, что пишем на языке «G». Формально такого языка не существует, но в этом и заключается прелесть этого средства разработки: от версии к версии в язык вводятся всё новые конструкции. Сложно представить, что в следующей реинкарнации Си появится, например, новая структура для for-цикла. А в LabVIEW такое вполне возможно.  
Впрочем надо заметить, что LabVIEW входит в рейтинг языков программирования [TIOBE](http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html), занимая на данный момент тридцатое место — где-то между Прологом и Фортраном.

**NI LabVIEW — история создания**

Компания National Instruments была создана в 1976 году тремя основателями — Джеффом Кодоски (Jeff Kodosky), Джеймсом Тручардом (James Truchard) и Биллом Новлиным (Bill Nowlin) в американском городе Остин (Austin), штат Техас. Основной специализацией компании являлись инструментальные средства для измерений и автоматизация производства.  
Первая версия LabVIEW увидела свет спустя десять лет после создания компании — в 1986 году (это была версия для Apple Mac). Инженеры NI решили бросить вызов «традиционным» языкам программирования и создали полностью графическую среду разработки. Основным идеологом графического подхода стал Джефф. Год за годом выпускались новые версии. Первой кроссплатформенной версией (включая Windows) была третья версия, выпущенная в 1993 году. Актуальной на данный момент является версия 8.6, вышедшая в прошлом году.  
  
В Остине и по сегодняшний день располагается головной офис компании. Сегодня в компании работают почти четыре тысячи человек, а офисы находятся почти в сорока странах (есть также офис и в России).

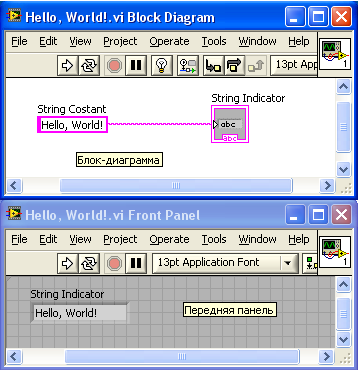
## Так что же такое LabVIEW?

LabVIEW — это кроссплатформенная графическая среда разработки приложений. LabVIEW — в принципе универсальный язык программирования. И хотя этот продукт порой тесно связан с аппаратным обеспечением National Instruments, он тем не менее не связан с конкретной машиной. Существуют версии для Windows, Linux, MacOS. Исходные тексты переносимы, а программы будут выглядеть одинаково во всех системах. Код, сгенерированный LabVIEW также может быть также исполнен на Windows Mobile или PalmOS (справедливости ради надо отметить, что поддержка PalmOS прекращена, впрочем здесь сама Palm больше виновата). Этот язык может с успехом использоваться для создания больших систем, для обработки текстов, изображений и работы с базами данных.  
  
LabVIEW — весьма высокоуровневый язык. Однако ничто не мешает включать «низкоуровневые» модули в LabVIEW-программы. Даже если вы хотите использовать ассемблерные вставки — это тоже возможно, надо лишь сгенерировать DLL и вставить вызовы в код. С другой стороны, высокоуровневый язык позволяет запросто производить весьма нетривиальные операции с данными, на которые в обычном языке могли уйти многие строки (если не десятки строк) кода. Впрочем, ради справедливости надо отметить, что некоторые операции низкоуровневых языков (например, работу с указателями), не так просто реализовать в LabVIEW ввиду его «высокоуровневости». Разумеется, язык LabVIEW включает основные конструкции управления, имеющие аналоги и в «традиционных» языках:

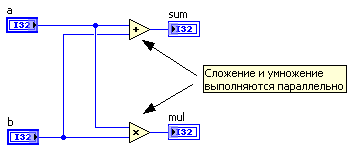
* переменные (локальные или глобальные)
* ветвление (case structure)
* For – циклы с проверкой завершения и без.
* While – циклы
* Группировка операций.

## LabVIEW – программа и возможности языка

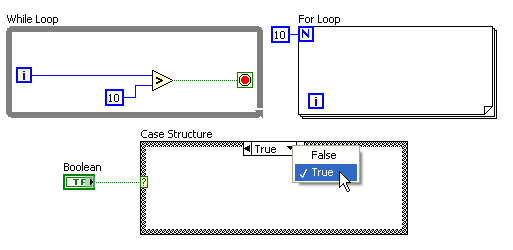
В LabVIEW разрабатываемые программные модули называются «Virtual Instruments» (Виртуальные Инструменты) или по-простому VI. Они сохраняются в файлах с расширением \*.vi. VIs – это кирпичики, из которых состоит LabVIEW – программа. Любая LabVIEW программа содержит как минимум один VI. В терминах языка Си можно достаточно смело провести аналогию с функцией с той лишь разницей, что в LabVIEW одна функция содержится в одном файле (можно также создавать библиотеки инструментов). Само собой разумеется, один VI может быть вызван из другого VI. В принципе каждый VI состоит из двух частей — Блок-Диаграмма (Block Diagram) и Передняя Панель (Front Panel). Блок-диаграмма — это программный код (точнее визуальное графическое представление кода), а Передняя панель — это интерфейс. Вот как выглядит классический пример Hello, World!:



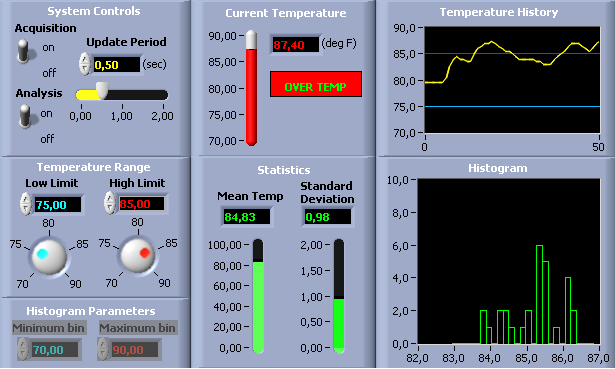
В основе LabVIEW лежит парадигма потоков данных. В вышеприведённом примере константа и терминал индикатора соединены между собой линией. Эта линия называется Wire. Можно назвать её «проводом». По проводам передаются данные от одних элементов другим. Вся эта концепция называется Data Flow. Суть Блок Диаграммы — это узлы (ноды), выходы одних узлов присоединены ко входам других узлов. Узел начнёт выполнение только тогда, когда прибудут все необходимые для работы данные. На диаграмме вверху две ноды. Одна из них — константа. Этот узел самодостаточен — он начинает выполнение немедленно. Второй узел — индикатор. Он отобразит данные, которые передаёт константа (но не сразу, а как только данные прибудут от константы).  
  
Вот чуть более сложный пример: сложение и умножение двух чисел. В традиционных языках мы напишем что-то вроде  
  
int a, b, sum, mul;  
//...  
sum = a + b;  
mul = a \* b;  
  
Вот как это выглядит в LabVIEW:



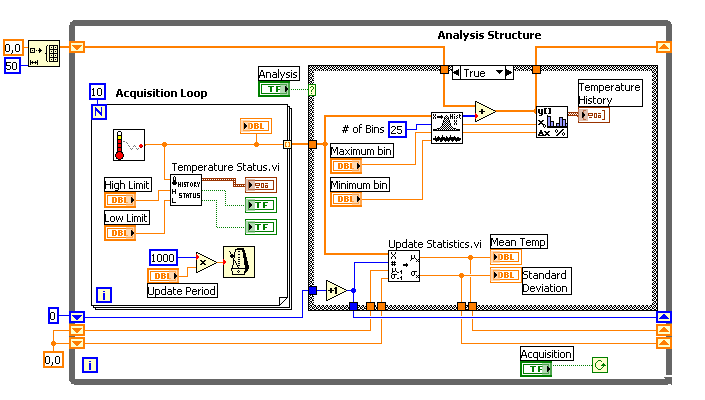
Обратите внимание на то, что сложение и умножение автоматически выполняются параллельно. На двухпроцессорной машине будут автоматически задействованы оба процессора.  
  
А вот как выглядят while / for циклы и if / then / else структура:



Как уже упоминалось, все элементы будут выполняться параллельно. Вам не нужно задумываться о том, как распараллелить задачу на несколько потоков, которые можно выполнять параллельно на нескольких процессорах. В последних версиях можно даже явно указать на каком из процессоров должен выполняться тот или иной while-цикл. Сейчас существуют надстройки и для текстовых языков, позволяющие запросто добиться поддержки многопроцессорных систем, однако так просто, как на LabVIEW, это пожалуй нигде не реализовано. (ну вот, я всё же скатился на сравнение с текстовыми языками). Если уж мы заговорили о многопоточности, то надо также отметить, что в распоряжении разработчика богатый выбор инструментов для синхронизации потоков — семафоры, очереди, рандеву, и т.д.  
  
LabVIEW включает в себя богатые наборы элементов для построения пользовательских интерфейсов. Уж на что быстро «набрасывались» интерфейсы в Дельфи, а в LabVIEW этот процесс происходит ещё стремительнее.



Стандартная поставка LabVIEW включает в себя также блоки для работы с ini файлами, реестром, функции для работы с двоичными и тестовыми файлами, математические функции, мощные инструменты для построения графиков (а куда же без этого в лаборатории-то), а в дополнение к уже упомянутой возможности вызовов DLL, LabVIEW позволяет работать с ActiveX компонентами и .net. Начиная с восьмой версии в LabVIEW была добавлена поддержка классов — язык стал объектно-ориентированным. Реализованную поддержку нельзя назвать полной, однако основные черты объектно-ориентированных языков — наследование и полиморфизм присутствуют. Также функциональность языка можно расширить дополнительными модулями, например NI Vision Toolkit – для обработки изображений и машинного зрения и другие. А при помощи модуля Applcation Builder можно сгенерировать исполняемый exe-файл. С помощью [Internet Toolkit](http://sine.ni.com/nips/cds/view/p/lang/en/nid/2501) можно работать с ftp серверами, c помощью [Database Connectivity Toolkit](http://sine.ni.com/nips/cds/view/p/lang/en/nid/6429) — с базами данных и т.д.



**Уроки по LabView**

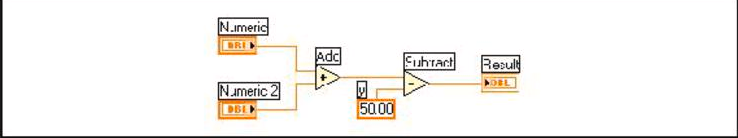
[**http://www.picad.com.ua/lesson.htm**](http://www.picad.com.ua/lesson.htm)

**Краткое изложение теоретического материала**

* ВП состоит из четырех основных компонентов – лицевой панели, блок-диаграммы, иконки и соединительной панели.
* Лицевая панель – интерфейс пользователя ВП.
* Блок-диаграмма – графический исходный текст программы, состоящий из узлов, терминалов данных и проводников.
* Палитра инструментов предназначена для создания и редактирования ВП. Удерживая нажатой клавишу **<Shift>** и щелкнув правой клавишей мыши, можно вывести на экран временную версию Палитры Инструментов.
* Палитра **Controls**(Элементов)предназначена для создания интерфейса лицевой панели. Для вывода на экран палитры **Controls**(Элементов)следует щелкнуть правой кнопкой мыши в открытом пространстве лицевой панели.
* Палитра **Functions**(Функций)предназначена для создания блок-диаграммы. Для вывода на экран палитры **Functions** (Функций) следует щелкнуть правой кнопкой мыши в открытом пространстве блок-диаграммы.
* Все объекты LabVIEW, свободное рабочее пространство лицевой панели и блок-диаграммы имеют свое контекстное меню. Обращение к всплывающему меню производится щелчком правой кнопкой мыши на объекте, лицевой панели или блок-диаграмме.

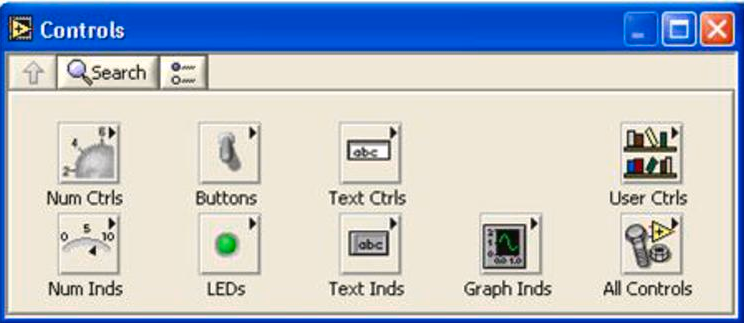
## Организация программной среды в LabVIEW

В Visual Basic, C++, Java и большинстве других текстовых языков программирования порядок выполнения всей программы определяется расположением функций программы.  
  
В среде LabVIEW используется потоковая модель обработки данных. Узлы блок-диаграммы выполняют заложенные в них функции, если данные поступили на все необходимые поля ввода/вывода. По окончании выполнения операции одним узлом результаты операции по проводникам данных передаются следующему узлу и т.д. Другими словами, готовность входных данных определяет последовательность выполнения узлов блок-диаграммы.  
  
В качестве примера можно рассмотреть блок-диаграмму, которая складывает два числа и затем вычитает из получившейся суммы «50.0». В этом случае блок-диаграмма выполняется слева направо не потому, что объекты помещены в этом порядке, а потому, что одно из полей ввода функции Subtract (Вычитание) не определено, пока не выполнилась функция Add (Сложение) и не передала данные к функции Subtract (Вычитание). Не следует забывать, что узел выполняется только тогда, когда определены его поля ввода данных.



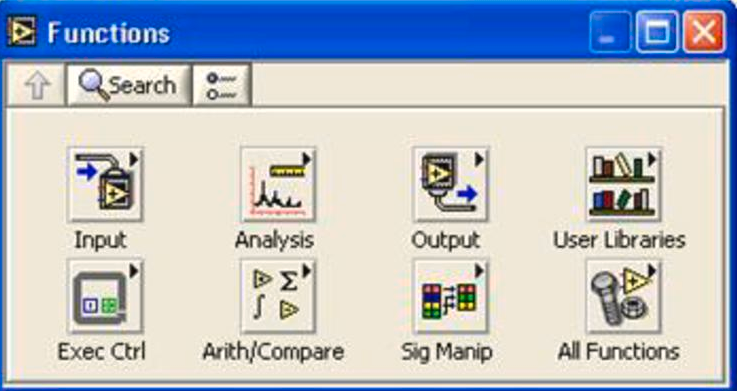
## Палитра Элементов

Палитра Элементов используется для размещения элементов управления и отображения на лицевой панели. Она доступна только на лицевой панели. Чтобы отобразить палитру Элементов, следует либо выбрать в пункте главного меню Window»Show Controls Palette, либо щелкнуть правой кнопкой мыши в рабочем пространстве лицевой панели. Используя кнопку в верхнем левом углу палитры, можно зафиксировать ее на экране. По умолчанию палитра Элементов появляется в экспресс-виде, показанном ниже, и содержит лишь наиболее часто используемые элементы.  
  
Используя кнопку All Controls, находящуюся в правом нижнем углу, можно разместить все элементы на палитре. Такая палитра, является исходной палитрой Элементов для всех упражнений этого курса.



## Палитра Функций

Палитра Функций используется для создания блок-диаграммы. Она доступна только на блок-диаграмме. Чтобы отобразить палитру Функций, следует либо выбрать в пункте главного меню Window»Show Functions Palette, либо щелкнуть правой кнопкой мыши в рабочем пространстве блок-диаграммы. Используя кнопку в верхнем левом углу палитры можно зафиксировать ее на экране. По умолчанию палитра Функций появляется в экспресс-виде и отображает экспресс-ВП. Экспресс-ВП — узлы функций, которые можно настраивать с помощью диалогового окна. Они используются для выполнения стандартных измерений при минимальных соединениях.  
При нажатии кнопки Options (опции), показанной слева, отображается страница Controls/Functions Palettes диалогового окна Options. Следует заменить Palette View на Advanced. Такая палитра, является исходной палитрой Функций для всех упражнений этого курса.



**Интернет-ресурсы**

* [**https://media.ls.urfu.ru/510/1322/2966/**](https://media.ls.urfu.ru/510/1322/2966/)
* [**https://ru.wikipedia.org/wiki/LabVIEW**](https://ru.wikipedia.org/wiki/LabVIEW)
* [**https://habr.com/ru/post/57859/**](https://habr.com/ru/post/57859/)
* [**https://engineering-solutions.ru/products/ni/labview/**](https://engineering-solutions.ru/products/ni/labview/)
* [**https://www.ni.com/ru-ru/shop/labview.html**](https://www.ni.com/ru-ru/shop/labview.html)

\