LabVIEW 术语快速索引

LabVIEW 模板：

- ◆ 工具模板（Tools Palette）
- ◆ 控件模板（Controls Palette）
- ◆ 功能模板（Functions Palette）

VI 的组成：

- ◆ **前面板（Panel）**

  控制（Control），指示（Indicator），修饰（Decoration）。

  将前面板中的控制和指示统称为前面板对象或控件。

- ◆ **框图程序（Diagram Programme）**

  节点（Node），数据连线（Wire）

  节点有：功能函数(Functions)，结构(Structures)，代码接口节点(CIN)，子 VI(SubVI)。

  数据端口有：控制端口和指示端口，节点端口。

  LabVIEW 编程又称为“数据流编程”。

- ◆ **图标/连接端口（Icon/Terminal）**

  把 VI 作为一个 SubVI 在其它 VI 中调用。

常用术语 ：

SubVI	子 VI	Chart	实时趋势图
LLBs	VI 库	Graph	事后记录图
Objects	对象	Functions	功能
Panel	前面板	Structures	结构
Block Diagram	框图程序	Cluster	簇
Control	控制	Bundle	打包
Indicator	指示	Unbundle	解包
Control 和 Indicator	前面板对象或控件	RefNum	枚举，标志号
Palette	模板	Local Variable	本地变量
Functions Palette	功能模板	Global Variable	全局变量
Controls Palette	控件模板	Constant	常量
Tools Palette	工具模板	Disable Indexing	无索引
Terminal	端口	Enable Indexing	有索引
Wires	数据连线	Read Local	本地读
Bad Wires	错误数据连线	Write Local	本地写
Node	节点	Read Global	全局读
Attribute Node		Write Global	全局写
Property Node	属性节点	Legend	图例
Frame	框架	Cursor	光标
Channel	框架通道	Bounds	边界范围
Index	索引	Data Acqisition(DAQ)	数据采集
Shift Register	移位寄存器	Label	标签

运行 VI

1. 运行 VI(Run)
2. 连续运行 VI(Run Continuously)
3. 停止运行 VI(Abort Execution)
4. 暂停运行 VI(Pause)

调试 VI

1. 单步执行
单步（入），单步（跳），单步（出）
2. 设置端点
3. 设置探针
4. 显示数据流动画

数据类型：

基本数据类型：数字型（Numeric），布尔型（Boolean），字符串型（String）
构造数据类型：数组（Array），簇（Cluster）
其它数据类型：枚举（RefNum），空类型

数组（Array）：

索引号从 0 开始
一维数组（1D，列或向量），二维数组（2D，矩阵）
组成：数据类型，数据索引（Index），数据
创建：1.控制模板->Array & Cluster 子模板
2.根据需要将相应数据类型的前面板对象放入数组框架中
使用：

1. Array Size 返回输入数组的长度
2. Index Array 返回输入数组由输入索引指定的元素
3. Replace Array Element 替换输入数组的一个元素
4. Array Subset 从输入数组取出指定的元素
5. Reshape Array 改变输入数组的维数
6. Initialize Array 初始化数组
7. Build Array 建立一个新数组
8. Rotate 1D Array 将输入数组的最后 n 个元素移至数组的最前面
9. Sort 1D Array 将数组按升序排列
10. Reverse 1D Array 将输入的 1D 数组前后颠倒，输入数组可以是任何类型的数组
11. Transpose 2D Array 转置输入的二维数组，也叫矩阵转置
12. Search 1D Array 搜索指定元素在一维数组中的位置
13. Array Max & Min 返回输入数组中的最大值和最小值
14. Split 1D Array 将输入的一维数组在指定的元素处截断，分成 2 个一维数组
15. Interpolate 1D Array 线性插值
16. Threshold 1D Array 一维数组阀值，是线性插值的逆过程
17. Interleave 1D Arrays 将从输入端口输入的一维数组插入到输出的一维数组中
18. Decimate 1D Array 将输入的一维数组分成数个一维数组，是 Interleave 1D Arrays 的逆过程

簇（Cluster）：

类似于 Pascal 语言的 record 和 C 语言的 struct

组成：不同的数据类型
创建：控制面板—>Array & Cluster 子面板；向框架添加所需的元素；根据需要更改簇和簇中元素的名称
使用：
1．Unbundle 解包。获得簇中元素的值
2．Bundle 打包。将相互关联的不同数据类型的数据组成一个簇，或给簇中的某个元素赋值
3．Unbundle By Name 按名称解包。获得由元素名称指定簇中相应元素的值
4．Bundle By Name 按名称打包。将相互关联的不同数据类型的数据组成一个簇，或给簇中的某个元素赋值
5．Build Cluster Array 建立簇的数组
6．Index & Bundle Cluster Array 将输入数组的元素按照索引组成簇，然后将这些簇组成一个数组
7．Cluster To Array 将簇转化为数组
8．Array To Cluster 将数组转化为簇

结构

For 循环（For Loop）

For（i=0；i<N；i++）
{
}
功能模板—>Structure 子模板
组成：循环框架（Loop Frame）：内有节点
重复端口（Iteration Terminal）：N
计数端口（Count Terminal）：i，初值为 0，递增步长为 1
移位寄存器（Shift Register）：右侧移位寄存器（第 i-1 次）—>左侧移位寄存器（第 i 次）
框架通道（Channel）：循环开始前，循环外节点—>循环内节点
循环结束时，循环内节点—>循环外节点
索引（Enable Indexing）—>数组
无索引（Disable Indexing）—> 最后一个数
自动索引（Auto Indexing）：循环执行时自动检测数组长度，并在每次循环时将数组中的元素按顺序一一取出

While 循环

当循环次数不能确定时，用 While 循环
while（条件）
{
}
do
{
} while（条件）
组成：循环框架（Loop Frame）
重复端口（Iteration Terminal）
条件端口（Conditional Terminal）每次循环结束时，条件端口检测数据连线输入的布尔值，若为 TRUE，停止循环；若为 FALSE，继续循环。如果不赋值，只执行一次

移位寄存器（Shift Register）
框架通道（Channel）

顺序结构（Sequence Structure）
•传统编程语言：控制流程（Control Flow）
LabVIEW：　　数据流程（Data Flow）
在 LabVIEW 中只有当某个节点的所有输入均有效时，LabVIEW 才能执行该节点
—>数据从属性（Data Dependency）
•**组成**：顺序框架（Sequence Frame）
　　框图标识符（Diagram Identifier）
　　递增/递减按扭（Increment/Decrement Buttons）
•**本地结果**（Sequence Local）：在顺序框架中向后传递数据
•**框架通道**（Frame Channel）：无 Enable Indexing 和 Disable Indexing 两种属性
•**公共连线**（Common Threads）：建立流程控制权（Flow Control Right）
　　Error Cluster 也是一种很好的公共连线，这种技术称为 ERROR I/O

选择结构（Case Structure）
switch（表达式）
{case　常量表达式 1：语句 1；
case　常量表达式 2：语句 2；
　　⋮
case　常量表达式 n：语句 n；
default　　：语句 n+1;
}

if（条件判断表达式）
{
}
else
{
}
组成：选择框架（Case Frame）
　　选择端口（Selection Terminal）：布尔型，数字整型，字符串型
　　框图标识符（Diagram Identifier）
　　递增/递减按扭（Increment/Decrement Buttons）

公式节点（Formula Node）
创建：1．功能模板—>Structures 子模板—>Formula Node
　　2．添加输入输出端口
　　3．按照 C 语言的语法规则在公式节点的框架中加入程序代码

属性节点（Attribute Node）
改善人机交互界面

使用：1.Visible Attribute

2.Disabled Attribute

3.Key Focus Attribute

4.Blinking Attribute

5.Position Attribute

6.Bounds Attribute（Read Only）

//

波形显示控件

按照处理测量数据的方式和显示过程的不同，LabVIEW 波形显示控件主要分为两类：

事后记录图（Graph）：基本数据结构是数组，一次显示完成

实时趋势图（Chart）：基本数据结构是数据标量或数组，连续不断显示

数据采集（Data Acquisition,DAQ）

一般而言，所有能够在计算机控制下完成数据采集和控制任务的板卡产品都称为 **DAQ 产品**。其分为**内插式**（plug-in）板卡和**外挂式**板卡。内插式 DAQ 板卡包括基于 ISA，PCI，PXI/Compact PCI，PCMCIA 等总线的板卡，速度快，但插拔不方便；外挂式 DAQ 板卡包括 USB，IEEE1394，RS232/RS485 和并口板卡，使用方便，但速度相对较慢。

NI 公司对其全部 DAQ 产品提供了专门的驱动程序库。NI 公司的 LabWindows/CVI 是开发 LabVIEW 驱动程序的最好工具。

DAQ 基本功能

◆A/D

将模拟量输入转换为数字量，是 DAQ 的核心。就工作原理而言，有逐次逼近法 A/D，双积分法 A/D，并行比较法 A/D。指标：采样分辨率，即 A/D 转换位数；A/D 转换速度。

◆D/A

将数字量转换为模拟量输出。指标：分辨率；线性误差

◆**数字 I/O（Digital I/O）**

与外部通信。一般采用 TTL 电平。

◆**定时（Timer）/计数（Counter）器**

脉冲周期信号测量，精确时间控制，脉冲信号产生。指标：分辨率；时钟频率。

DAQ VIs 的位置

功能模板—>Data Acquisition 子模板（ 6 个）

◆Analog Input

◆Analog Output

◆Digital I/O

◆Counter

◆Calibration and Configuration 校准和配置 DAQ 设备，并返回 DAQ 设备的配置信息

◆Signal Conditioning 将采集到的模拟电压信号转换为相应的应力单位或温度单位

DAQ VIs 的组织结构