TickFormat 축의 눈금값 형식

디폴트 설정

IDL이 그래픽을 생성할 때, 값의 범위를 균등 분할 하여 눈금을 생성하고 그 값을 표시합니다. 상황에 따라 포맷이 조금 달라집니다. 일단 그려 보고 마음에 들지 않으면 아래 설정 방법들을 검토해 보세요.

```
plot, indgen(10)        ;Direct Graphics
p=plot(indgen(10))      ;New Graphics
```

축 눈금의 직접 지정

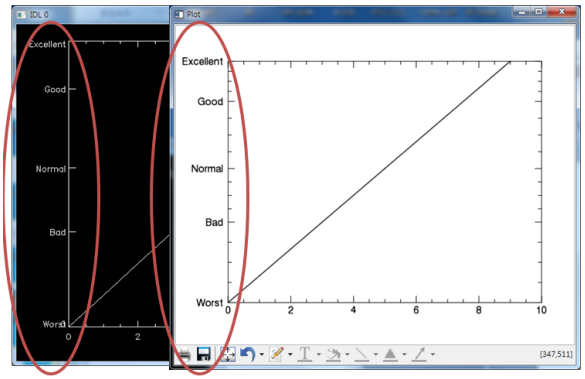
다음과 같이 축 눈금을 표시할 값(위치)과 표출내용을 자유롭게 지정할 수 있습니다. 상황에 따라 유일한 방법일 때도 있으며, 인쇄본 제출 마감 전날 같이 급박한 경우 가장 유용한 방법일 수 있습니다.

```
plot, indgen(10), YTICKV=[0, 3, 5, 7.5, 9], YTICKS=6, $
  YTICKNAME=['Worst', 'Bad', 'Normal', 'Good', 'Excellent']
```

다이렉트 그래픽스에서는 어느 값에 눈금을 쓸 것인지(YTICKV)와 눈금의 개수(YTICKS)가 잘 맞아야 정상적인 표출이 됩니다. 이 경우 YTICKV의 개수가 N개라면, YTICKS의 개수가 N+1개여야 합니다. 그리고 N개의 YTICKNAME을 지정해 주면 지정한 문자열이 축 눈금값으로 표출됩니다.

```
p=plot(indgen(10), YTICKVALUES=[0, 3, 5, 7.5, 9], $
   YTICKNAME=['Worst', 'Bad', 'Normal', 'Good', 'Excellent'])
```

New 그래픽스에서는 어느값에 눈금을 쓸 것인지(YTICKVALUES)와 해당 값에 대한 표출 내용(YTICKNAME)을 써 주는 것으로 눈금값 을 직접 지정합니다. 가끔 다이렉트 그래픽스의 키워드와 New 그래픽스의 키워드가 같은 내용인데 이름이 다른 경우가 있습니다(위 예제에서 YTICKV와 YTICKVALUES가 그런 경우입니다). 축을 설정하는 이런 류의 키워드는 모두 XTICK...., YTICK...., ZTICK.... 과 같이 세 축 모두 적용 가능합니다. [XYZ]TICK... 로 표현하기도 합니다.



다이렉트 그래픽스와 New 그래픽스에서 축 눈금을 직접 지정

표준 포맷 문자열(Format Code) 사용

[XYZ]TICKFORMAT 키워드를 사용하여, 숫자를 변환하는 표준 문자열 포맷을 사용할 수 있습니다.

```
x=findgen(10)-4
y=10.^x
plot, x, y, psym=-2, /YLOG
plot, x, y, psym=-2, /YLOG, XTICKFORMAT='(F5.1)', $
   YTICKFORMAT='(E8.1)'
p=plot(x, y, '-*', /YLOG)
p=plot(x, y, '-*', /YLOG, XTICKFORMAT='(F5.1)', $
   YTICKFORMAT='(E8.1)')
```

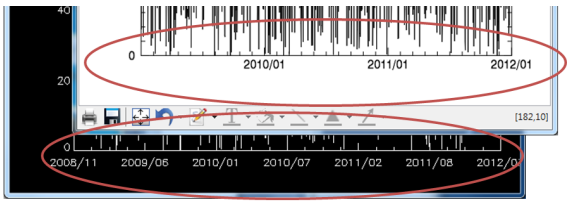
디폴트 설정의 눈금표출과 TICKFORMAT을 지정한 눈금표출을 비교해 보세요. 표준 포맷 코드를 사용할 때는 괄호로 감싼다는 것을 기억하세요. 이 방식을 사용하여 간단하게 $를 앞에 붙인다든지, %를 뒤에 붙이는 등의 일도 할 수 있습니다.

```
plot, indgen(100), ytickformat='(I3,"%")'
p=plot(indgen(100), ytickformat='(I3, "%")')
```

날짜와 시간 축(표준 포맷 코드 사용)

표준 포맷 코드에는 날짜체계 표출을 위한 C( ) 코드가 있습니다. 줄리언 날짜 값이라면 C( ) 코드를 이용하여 날짜로 표출하는 것이 간단합니다.

```
x=timegen(start=julday(1,1,2009), final=julday(12,31,2011),  $
  UNITS='DAYS')
ntms=n_elements(x)
y=randomu(-1L, ntms)*100
plot, x, y, xtickformat='(C(CYI04,"/",CMOI02))'
p=plot(x, y, xtickformat='(C(CYI04, "/", CMOI02))')
```



도움말에서 FORMAT CODES로 검색해 보면 어떤 종류의 포맷 코드를 사용할 수 있는지 확인할 수 있습니다. 다양한 옵션이 있어 대부분은 이 안에서 해결할 수 있을 것입니다.

날짜와 시간 축(LABEL_DATE 사용)

LABEL_DATE( ) 함수를 사용하면 좀 더 간단하고 다양하게 날짜/시간 축을 설정할 수 있습니다. 앞의 예제를 LABEL_DATE()로는 다음과 같이 수행합니다.

```
dummy=label_date(date_format='%Y/%N')
plot, x, y, xtickformat='LABEL_DATE'
p=plot(x, y, xtickformat='LABEL_DATE')
```

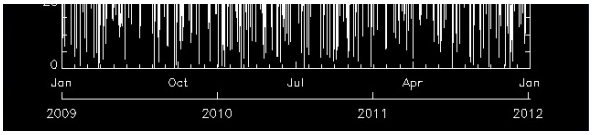
TICKFORMAT에 주어지는 문자열에 괄호가 없다는 것을 눈여겨보십시오. 도움말에서 LABEL_DATE 함수를 검색해 보면 date_format으로 사용하는 코드를 확인해 볼

수 있습니다. 게다가 LABEL_DATE 함수에 AM_PM=, DAYS_OF_WEEK=, MONTHS= 키워드가 있어 [오전,오후], [요일], [월이름]을 마음대로 지정할 수도 있습니다. 이러한 변환은 모두 "줄리언 날짜" 체계를 이용한다는 점을 기억해 두십시오.

축 눈금의 Level(다중 축)

앞의 예제에서는 [XYZ]TICKFORMAT 키워드의 값을 단일값으로 주었습니다만, 두 개 이상의 문자열 배열로 설정하는 것이 가능합니다. 다중축을 생성하겠다는 의도인데 [XYZ]TICKUNITS를 함께 사용하여 각 축의 단위도 지정해 주어야 합니다. 말이 복잡하지 어렵지 않을 것입니다. 다음 예제를 보세요.

```
tickformat=['(C(CMoA))', '(C(CYI04))']
tickunit=['Months', 'Years']
plot, x, y, xtickformat=tickformat, xtickunit=tickunit, $
  position=[0.1, 0.2, 0.9, 0.9]
```



X축이 월 단위(Months)와 연단위(Years)의 두 개의 축으로 구성되었습니다. 축이 차지하는 공간을 고려해 POSITION 키워드와 같이 위치설정도 해 줄 필요가 있습니다. 이럴 때 TICKFORMAT도 TICKUNIT와 같은 개수로 설정을 합니다. 위 예제에서는 0레벨 포맷이 (C(CMoA))로 월이름이고 1레벨 포맷이 (C(CYI04))로 연도 숫자 네자리입니다. TICKUNITS 키워드에 줄 수 있는 값으로 'Numeric', 'Years', 'Months', 'Days', 'Hours', 'Minutes'. 'Seconds', 'Time'이 있습니다. 자세한 내용은 도움말을 보십시오.

LABEL_DATE를 이용하여도 다단계 축을 그려낼 수 있습니다.

```
dummy=label_date(date_format=["%M", "'%Z"])
plot, x, y, xtickformat='LABEL_DATE', $
  xtickunit=['Months', 'Years'], position=[0.1, 0.2, 0.9, 0.9]
```

New 그래픽스는 그래픽 컴포넌트 추가가 가능한 체계이므로 AXIS() 함수를 이용하여 축을 추가하는 방법이 있으며 더욱 다양한 설정이 가능합니다.

사용자가 작성한 함수로 눈금값 표출

포맷 변환 코드로 원하는 결과를 얻지 못할 때, 맨 앞에 소개한 눈금값 직접 설정 방법을 검토해 보고 이로서도 만족스럽지 않다면 눈금값 변환 함수를 직접 만들 수 있습니다. 함수 안에서 사용자는 다양한 조건문을 사용할 수도 있고, 원하는 어떤 복잡한 설정도 할 수 있습니다.

- 함수의 이름은 자유롭습니다.
- 함수가 받을 인자는 3개로 모두 그림을 그리는 명령에서 함수의 인자로 값을 넣어줄 것입니다. 어떤 값을 받게 될지 알아야 함수를 설계하겠지요?
- 첫 인자는 축의 종류(axis)입니다. 0은 X축, 1은 Y축, 2는 Z축입니다. YTICKFORMAT 키워드로 사용자 함수를 호출한다면? 예, 1이 자동으로 넘겨집니다.
- 두 번째 인자는 눈금의 인덱스입니다(0으로 시작).
- 세 번째 인자는 눈금의 값입니다(Double형으로 전달).
- 만일, [XYZ]TICKUNITS 키워드로 두 개 이상의 단위를 사용한다면, 다중 축이 생성되고 축의 레벨이 있게 됩니다. 이 경우 사용자 함수는 네 번째 인자를 받게 되며 0으로 시작하는 축의 레벨을 의미합니다.

IDL 그래픽의 이런 체계를 이용하여 사용자가 함수를 만들어, 축의 종류에 따라, 축 눈금의 순서(index)에 따라, 눈금 값에 따라, 다중 축의 레벨에 따라 자유롭게 눈금값을 출력하도록 지정할 수 있습니다. 처음에는 말보다 예제가 더 이해하기 좋습니다.

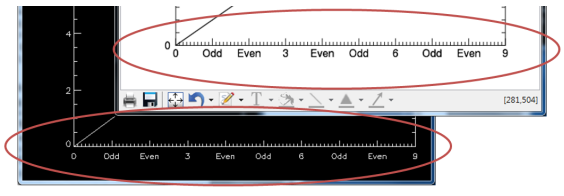
[예제] 눈금값이 짝수면 'Even'을, 홀수면 'Odd'를 눈금값으로 출력해야 합니다. 만일 눈금값이 3의 배수면 그 값을 숫자로 그대로 출력합니다. 이런 독특한 눈금체계를 사용하고자 할 때, 다음과 같은 함수를 만들지요.

```
function mytick, axis, index, value
; print, axis, index, value
intval=round(value)
if intval mod 2 eq 0 then ret='Even' else ret='Odd'
if intval mod 3 eq 0 then ret=string(value, format='(I)')
return, ret
end
```

사용자 함수를 활용하는 방법은 다음과 같습니다.

```
plot, indgen(10), xtickformat='mytick', xtickinterval=1
p=plot(indgen(10), xtickformat='mytick', xtickinterval=1)
```

사용자 함수로 어떤 값들이 넘어오는지 확인해 보고자 한다면 mytick 함수 두 번째 줄의 주석 처리를 해제하



고 값들을 print 해 보세요. 별 것 없습니다.

1. 설정이 xtickinterval=1 이므로 0~9까지 1씩 증가하는 축 눈금이 됩니다.
2. xtickformat='mytick'에서 X축을 위해 호출되었으므로 첫 인수는 0(X축)이 전달됩니다.
3. 순서대로 축 눈금값(0.0, 1.0, 2.0, ..., 9.0)을 받아 실제 출력될 내용(문자열)으로 변환하여 return 합니다.