ACTIVIDAD 9. GUI's y orígenes de datos.

Vamos a realizar un ejemplo que resume lo visto hasta ahora y que podría ser un modelo de supuesto práctico de examen.

Supongamos que nos piden un módulo para llevar a cabo el control en la recepción de coches en un taller dentro de una aplicación de gestión más amplia. El usuario de ese módulo debe *introducir, modificar, listar y eliminar* los coches que se recepcionen guardando los datos siguientes:

- **dni_cliente:** debe comprobar su validez
- **matrícula:** usaremos un entry
- **modelo:** uno o dos entry si queremos poner marca y modelo.
- **color:** un combo con 8-10 colores
- **fecha de entrada y fecha de salida:** *ver códigos de ayuda ya vistos*
- **observaciones:** un textArea donde el jefe de taller pueda incluir todo aquello que considere oportuno como estado del coche en el momento de recepción, valor cuentakm, etc..
- **tipo de reparación:** las opciones serían *sección de chapa y pintura, sección mecánica de motor, y sección de mantenimiento rápido (cambio de aceite, ruedas, aceite, ...),* para eso usaremos un checkbox*, ver códigos de ayuda*
- **aceptación de presupuesto:** un option con las opciones Sí o No.

En el listado del treeView solo es necesario que se cargue solamente el **dni_cliente, matrícula y fecha entrada y tipo de reparación.** Eso sí marcando uno cualquiera toda la información de ese coche se cargaría en los widgtes correspondientes. No es neceserio duplicarlos.

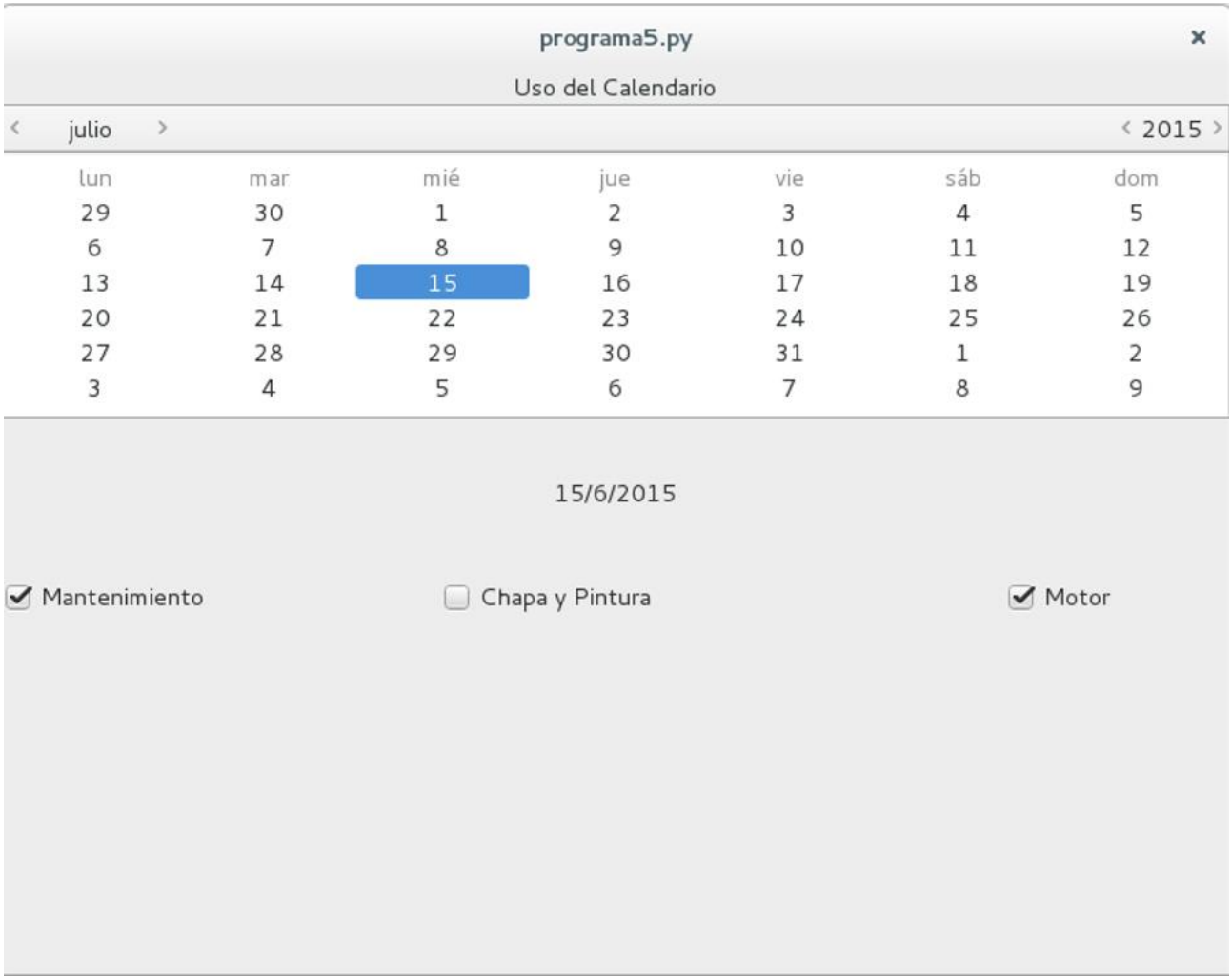
→ Códigos de ayuda

Uno de los aspectos habituales en las interfaces gráficas son los calendarios para cargar y seleccionar las fechas. Python tiene una función llamada Calendar. En el siguiente **enlace** podrás ver todas sus funciones. Código ejemplo:

```
def on_calendar_day_selected(self,widget, data=None):
    agno, mes, dia = self.calendario.get_date()
    self.fecha = "%s/" %dia + "%s/" %mes + "%s" %agno
    self.label2.set_text(self.fecha)
```

El otro código son los checkbox. Imaginemos que un cliente además de chapa y pintura precisa arreglo de motor y/o mantenimiento. El código del checkbox que os puede ayudar:

```
def on_chkmotor_toggled(self, widget):
  if self.chkmotor.get_active():
    print "motor"
```



El diseño del módulo es totalmente libre pero debe recoger toda la información dada por el cliente. Uno de los aspecto que se pueden mejorar sería el control de un cliente que repita algún servicio del taller en recepciones futuras ¿Cómo controlaríamos eso?