GoF João A.

Builder ([GPCalanderBase.java](https://github.com/anamfrancisco/ganttproject_SE/blob/master/biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/GPCalendarBase.java))

A classe não possui um construtor para inicializar as variáveis, em vez disso depende de métodos para tal.

abstract class GPCalendarBase implements GPCalendarCalc {

  private final List<GPCalendarListener> myListeners = Lists.newArrayList();

  private String myName;

 private String myId;

  @Override

  public String getID() {

    return myId == null ? myName : myId;

  }

  @Override

  public String getName() {

    return myName;

  }

  @Override

  public void setName(String name) {

    myName = name;

  }

  @Override

  public void setID(String id) {

    myId = id;

  }

Facade ([TimeUnitImpl.java](https://github.com/anamfrancisco/ganttproject_SE/blob/master/biz.ganttproject.core/src/main/java/biz/ganttproject/core/time/TimeUnitImpl.java))

A classe possui variáveis que representam instâncias de partes mais complexas dos sistemas, assim simplificando o seu uso.

public class TimeUnitImpl implements TimeUnit {

  private final String myName;

  private final TimeUnitGraph myGraph;

  private final TimeUnit myDirectAtomUnit;

  public TimeUnitImpl(String name, TimeUnitGraph graph, TimeUnit directAtomUnit) {

    myName = name;

    myGraph = graph;

    myDirectAtomUnit = directAtomUnit;

  }

State ([WeekendCalendarImpl.java](https://github.com/anamfrancisco/ganttproject_SE/blob/master/biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/WeekendCalendarImpl.java))

O método comporta-se de maneira diferente de acordo com o estado(que neste caso é o tipo de dia).

private boolean isPublicHoliDay(Date curDayStart) {

    CalendarEvent oneOff = myOneOffEvents.get(curDayStart);

    if (oneOff != null) {

      switch (oneOff.getType()) {

      case HOLIDAY:

        return true;

      case WORKING\_DAY:

        return false;

      case NEUTRAL:

      default:

        // intentionally fall-through, consult recurring holidays in this case

      }

    }

    CalendarEvent recurring = myRecurringEvents.get(getRecurringDate(curDayStart));

    if (recurring != null) {

      switch (recurring.getType()) {

      case HOLIDAY:

        return true;

      case WORKING\_DAY:

        return false;

      case NEUTRAL:

      default:

        // intentionally fall-through, use default answer

      }

    }

    return false;

  }