Classe Inútil (WebStartIDClass)

[WebStartIDClass.java](https://github.com/MiguelGoulao/ganttproject/blob/master/biz.ganttproject.impex.msproject2/src/main/java/biz/ganttproject/impex/msproject2/WebStartIDClass.java)

public class WebStartIDClass {

}

Método Longo (ProjectFileImporter)

[ProjectFileImporter.java](https://github.com/MiguelGoulao/ganttproject/blob/master/biz.ganttproject.impex.msproject2/src/main/java/biz/ganttproject/impex/msproject2/ProjectFileImporter.java)

private void importTask(Task t, net.sourceforge.ganttproject.task.Task supertask,

                       Map<Integer, GanttTask> foreignId2nativeTask, Map<GanttTask, Date> nativeTask2foreignStart) {

 if (t.getNull()) {

   myErrors.add(Pair.create(Level.INFO,

       MessageFormat.format("Task with id={0} is blank task. Skipped", foreignId(t))));

   return;

 }

 if (t.getUniqueID() == 0) {

   boolean isRealTask = t.getName() != null && !t.getChildTasks().isEmpty();

   if (!isRealTask) {

     for (Task child : t.getChildTasks()) {

       importTask(child, getTaskManager().getRootTask(), foreignId2nativeTask, nativeTask2foreignStart);

     }

     return;

   }

 }

 StringBuilder report = new StringBuilder();

 java.util.function.Function<Task, Pair<TimeDuration, TimeDuration>> getDuration = findDurationFunction(t, report);

 if (getDuration == null) {

   myErrors.add(Pair.create(Level.SEVERE,

       String.format("Can't determine the duration  of task %s (%s). Skipped", t, report)));

   return;

 }

 TaskBuilder taskBuilder = getTaskManager().newTaskBuilder()

     .withParent(supertask)

     .withName(t.getName())

     .withNotes(t.getNotes())

     .withWebLink(t.getHyperlink());

 if (t.getPriority() != null) {

   taskBuilder = taskBuilder.withPriority(convertPriority(t.getPriority()));

 }

 Date foreignStartDate = convertStartTime(t.getStart());

 if (t.getChildTasks().isEmpty()) {

   taskBuilder.withStartDate(foreignStartDate);

   if (t.getPercentageComplete() != null) {

     taskBuilder.withCompletion(t.getPercentageComplete().intValue());

   }

   if (t.getMilestone()) {

     taskBuilder.withLegacyMilestone();

   }

   Pair<TimeDuration, TimeDuration> durations = getDuration.apply(t);

   TimeDuration workingDuration = durations.first();

   TimeDuration nonWorkingDuration = durations.second();

   TimeDuration defaultDuration = myNativeProject.getTaskManager().createLength(

       myNativeProject.getTimeUnitStack().getDefaultTimeUnit(), 1.0f);

   if (!t.getMilestone()) {

     if (workingDuration.getLength() > 0) {

       taskBuilder.withDuration(workingDuration);

     } else if (nonWorkingDuration.getLength() > 0) {

       myErrors.add(Pair.create(Level.INFO, MessageFormat.format(

           "[FYI] Task with id={0}, name={1}, start date={2}, end date={3}, milestone={4} has working time={5} and non working time={6}.\n"

               + "We set its duration to {6}", foreignId(t), t.getName(), t.getStart(), t.getFinish(),

           t.getMilestone(), workingDuration, nonWorkingDuration)));

       taskBuilder.withDuration(nonWorkingDuration);

     } else {

       myErrors.add(Pair.create(Level.INFO, MessageFormat.format(

           "[FYI] Task with id={0}, name={1}, start date={2}, end date={3}, milestone={4} has working time={5} and non working time={6}.\n"

               + "We set its duration to default={7}", foreignId(t), t.getName(), t.getStart(), t.getFinish(),

           t.getMilestone(), workingDuration, nonWorkingDuration, defaultDuration)));

       taskBuilder.withDuration(defaultDuration);

     }

   } else {

     taskBuilder.withDuration(defaultDuration);

   }

 }

 GanttTask nativeTask = (GanttTask) taskBuilder.build();

 if (t.getCost() != null) {

   nativeTask.setCost(new CostStub(BigDecimal.valueOf(t.getCost().doubleValue()), false));

 }

 if (!t.getChildTasks().isEmpty()) {

   for (Task child : t.getChildTasks()) {

     importTask(child, nativeTask, foreignId2nativeTask, nativeTask2foreignStart);

   }

 }

 importCustomFields(t, nativeTask);

 foreignId2nativeTask.put(foreignId(t), nativeTask);

 nativeTask2foreignStart.put(nativeTask, foreignStartDate);

}

Dead Code (ProjectFileExporter)

[ProjectFileExporter.java](https://github.com/joaoLima17/ganttproject/blob/master/biz.ganttproject.impex.msproject2/src/main/java/biz/ganttproject/impex/msproject2/ProjectFileExporter.java)

private void exportTasks(Map<Integer, net.sf.mpxj.Task> id2mpxjTask) throws MPXJException {

//    Map<CustomPropertyDefinition, FieldType> customProperty\_fieldType = new HashMap<CustomPropertyDefinition, FieldType>();

//    collectCustomProperties(getTaskManager().getCustomPropertyManager(), customProperty\_fieldType, TaskField.class);

   Map<CustomPropertyDefinition, FieldType> customProperty\_fieldType = CustomPropertyMapping.buildMapping(getTaskManager());

   exportCustomFieldTypes(customProperty\_fieldType);

   net.sf.mpxj.Task rootTask = myOutputProject.addTask();

   rootTask.setEffortDriven(false);

   rootTask.setID(0);

   rootTask.setUniqueID(0);

   rootTask.setOutlineLevel(0);

   rootTask.setWBS("0");

   rootTask.setOutlineNumber("0");

   rootTask.setStart(convertStartTime(getTaskManager().getProjectStart()));

   rootTask.setFinish(convertFinishTime(getTaskManager().getProjectEnd()));

   rootTask.setDuration(convertDuration(getTaskManager().createLength(

       getTaskManager().getRootTask().getDuration().getTimeUnit(), getTaskManager().getProjectStart(),

       getTaskManager().getProjectEnd())));

   // rootTask.setDurationFormat(TimeUnit.DAYS);

   rootTask.setTaskMode(TaskMode.AUTO\_SCHEDULED);

   int i = 0;

   for (Task t : getTaskHierarchy().getNestedTasks(getTaskHierarchy().getRootTask())) {

     exportTask(t, null, 1, ++i, id2mpxjTask, customProperty\_fieldType);

   }

 }