* 1. 隐藏掉系统中的默认的菜单和工具项

public static void hideSystemMenu() {  
  IWorkbenchPage page = PlatformUI.getWorkbench().getActiveWorkbenchWindow().getActivePage();  
  page.hideActionSet("org.eclipse.ui.WorkingSetActionSet");  
  page.hideActionSet("org.eclipse.ui.edit.text.actionSet.annotationNavigation");  
  page.hideActionSet("org.eclipse.ui.edit.text.actionSet.navigation");  
  page.hideActionSet("org.eclipse.search.searchActionSet");  
 }

//把该方法在postWindowOpen()中调用 一下就ok了，如果是多个透视图的情况，在透视图转换后调用一下该方法就ok了

* 1. 透视图转换的操作：

public static void switchPerspective(String perspectiveID) {;//perspectiveID 是对应的透视图的ID，是自定义的  
  IPerspectiveRegistry reg = PlatformUI.getWorkbench().getPerspectiveRegistry();  
  IWorkbenchPage page = PlatformUI.getWorkbench().getActiveWorkbenchWindow().getActivePage();  
  page.setPerspective(reg.findPerspectiveWithId(perspectiveID));//从新设置透视图，根据透视图ID  
 //隐藏到系统的默认菜单项

  hideSystemMenu();  
 }

-----------------------------------

其实，隐藏菜单项和工具项也可以通过配置实现，而不是直接像上面硬编码的方式：

对org.eclipse.ui.perspectiveExtensions 透视图扩展点进行扩展：

在perspectiveExtension项下可以选择hiddenMenuItem、hiddenToolBarItem进行相应的隐藏操作。

 public void postWindowCreate()

 {

IActionBarConfigurer actionBarConfigurer = getWindowConfigurer().getActionBarConfigurer();

IContributionItem[] coolItems = actionBarConfigurer.getCoolBarManager() .getItems();

for (int i = 0; i < coolItems.length; i++)

{

if (coolItems[i] instanceof ToolBarContributionItem)

{

ToolBarContributionItem toolbarItem = (ToolBarContributionItem) coolItems[i];

if(toolbarItem.getId().equals("org.eclipse.ui.WorkingSetActionSet")|| toolbarItem .getId() .equals( org.eclipse.ui.edit.text.actionSet.annotationNavigation") ||

toolbarItem .getId() .equals( "org.eclipse.ui.edit.text.actionSet.navigation"))

{

toolbarItem.getToolBarManager().removeAll();

}

}

}

actionBarConfigurer.getCoolBarManager().update(true);

}

private static final String[] ACTIONSETID = new String[] {

"org.eclipse.ant.ui.actionSet.presentation", //$NON-NLS-1$  
         "org.eclipse.debug.ui.breakpointActionSet",//$NON-NLS-1$   
         "org.eclipse.debug.ui.debugActionSet",//$NON-NLS-1$   
         "org.eclipse.debug.ui.launchActionSet",//$NON-NLS-1$   
         "org.eclipse.debug.ui.profileActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.GoToPackage",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.GoToType", //$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.OpenExternalJavaDoc",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.OpenSuperImplementation", //$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.CopyQualifiedName",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.Open", //$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.OpenTypeHierarchy", //$NON-NLS-1$   
         "org.eclipse.jdt.ui.actions.OpenCallHierarchy",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.JavaElementCreationActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.JavaActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.A\_OpenActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.CodingActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.SearchActionSet",//$NON-NLS-1$   
         "org.eclipse.jdt.ui.text.java.actionSet.presentation",//$NON-NLS-1$   
         "org.eclipse.search.searchActionSet",//$NON-NLS-1$   
         "org.eclipse.team.ui.actionSet",//$NON-NLS-1$   
         "org.eclipse.ui.NavigateActionSet", //$NON-NLS-1$   
         "org.eclipse.ui.edit.text.actionSet.openExternalFile", //$NON-NLS-1$   
         "org.eclipse.ui.edit.text.actionSet.presentation",//$NON-NLS-1$   
         "org.eclipse.ui.edit.text.actionSet.annotationNavigation",//$NON-NLS-1$   
         "org.eclipse.ui.edit.text.actionSet.navigation",//$NON-NLS-1$   
         "org.eclipse.ui.externaltools.ExternalToolsSet",//$NON-NLS-1$   
         "org.eclipse.ui.edit.text.actionSet.convertLineDelimitersTo", //$NON-NLS-1$  
         "org.eclipse.ui.actionSet.keyBindings",//$NON-NLS-1$   
         "org.eclipse.ui.actionSet.openFiles",//$NON-NLS-1$   
         "org.eclipse.ui.WorkingSetActionSet",//$NON-NLS-1$   
         "org.eclipse.ui.WorkingSetModificationActionSet",//$NON-NLS-1$   
         "org.eclipse.ui.WorkingSetActionSet.toolbar",//$NON-NLS-1$   
         "org.eclipse.update.ui.softwareUpdates",//$NON-NLS-1$   
 };

 /\*  
  \* (non-Javadoc)  
  \*   
  \* @see org.eclipse.ui.application.ActionBarAdvisor#fillCoolBar(org.eclipse.jface.action.ICoolBarManager)  
  \*/  
 protected void fillCoolBar(ICoolBarManager coolBar) {  
     IToolBarManager toolbar = new ToolBarManager(SWT.FLAT | SWT.RIGHT);

     coolBar.add(new ToolBarContributionItem(toolbar, "save")); //$NON-NLS-1$

     // toolbar = new ToolBarManager(SWT.FLAT | SWT.RIGHT);  
     // coolBar.add(new ToolBarContributionItem(toolbar, "ads")); //$NON-NLS-1$  
     // AnnouncementContributionItem announcementItem = new AnnouncementContributionItem(Messages  
     // .getString("ApplicationActionBarAdvisor.message"));  
     // toolbar.add(announcementItem); //$NON-NLS-1$

     ActionSetRegistry reg = WorkbenchPlugin.getDefault().getActionSetRegistry();  
     IActionSetDescriptor[] actionSets = reg.getActionSets();  
     List list = Arrays.asList(ACTIONSETID);  
     for (int i = 0; i < actionSets.length; i++) {  
         if (list.contains(actionSets[i].getId())) {  
             removeAction(reg, actionSets[i]);  
         }  
     }  
 }

 private void removeAction(final ActionSetRegistry reg, final IActionSetDescriptor actionSet) {  
     IExtension ext = actionSet.getConfigurationElement().getDeclaringExtension();  
     reg.removeExtension(ext, new Object[] { actionSet });  
 }

* 1. 移除wizard中的General// 移除文件菜单下导入功能中自带的General

AbstractExtensionWizardRegistry importWizardRegistry = (AbstractExtensionWizardRegistry) WorkbenchPlugin

                 .getDefault().getImportWizardRegistry();

IWizardCategory[] categories = wizardRegistry.getRootCategory().getCategories();

for (int i = 0; i < categories.length; i++)

{

     if(categories[i].getId().equals("org.eclipse.ui.Basic"))

{

       IWizardDescriptor[] wizard = categories[i].getWizards();

       IExtensionRegistry registry = Platform.getExtensionRegistry();

       IExtensionPoint point = registry.getExtensionPoint("org.eclipse.ui.importWizards");

       IExtension[] extensions = point.getExtensions();

       for (int j = 0; j < wizard.length; j++)

{

           wizardRegistry.removeExtension(extensions[i], new Object[] {wizard[j] });

        }

      }

}

* 1. 移除wizard中的General// 移除文件菜单下导出功能中自带的General

AbstractExtensionWizardRegistry exportWizardRegistry = (AbstractExtensionWizardRegistry) WorkbenchPlugin.getDefault().getExportWizardRegistry();

IWizardCategory[] categories = wizardRegistry.getRootCategory().getCategories();

for (int i = 0; i < categories.length; i++)

{

   if(categories[i].getId().equals("org.eclipse.ui.Basic"))

{

    IWizardDescriptor[] wizard = categories[i].getWizards();

    IExtensionRegistry registry = Platform.getExtensionRegistry();

IExtensionPoint point = registry.getExtensionPoint("org.eclipse.ui.exportWizards");

    IExtension[] extensions = point.getExtensions();

for (int j = 0; j < wizard.length; j++)

{

        wizardRegistry.removeExtension(extensions[i], new Object[] {wizard[j] });

     }

    }

}

* 1. 移除view中的General// 移除窗口菜单下显示视图中自带的General

ViewRegistry viewRegistry = (ViewRegistry) WorkbenchPlugin.getDefault().getViewRegistry();

IViewCategory[] viewCat = viewRegistry.getCategories();

for (int i = 0; i < viewCat.length; i++)

{

if(viewCat[i].getId().equals("org.eclipse.ui"))

{

      IViewDescriptor[] viewDes = viewCat[i].getViews();

      IExtensionRegistry registry = Platform.getExtensionRegistry();

      IExtensionPoint point = registry.getExtensionPoint("org.eclipse.ui.views");

      IExtension[] extensions = point.getExtensions();

      for (int j = 0; j < viewDes.length; j++)

{

         if(!viewDes[j].getId().equals("org.eclipse.ui.console.ConsoleView"))

{

            viewRegistry.removeExtension(extensions[i], new Object[] {viewDes[j] });

            }

          }

   }

}

* 1. 移除ActionSet中的General

ActionSetRegistry reg = WorkbenchPlugin.getDefault().getActionSetRegistry();  
IActionSetDescriptor[] actionSets = reg.getActionSets();  
// removing annoying gotoLastPosition Message.  
String actionSetId = "org.eclipse.ui.edit.text.actionSet.navigation";   
for (int i = 0; i <actionSets.length; i++)  
{  
    if (!actionSets[i].getId().equals(actionSetId))  
        continue;  
    IExtension ext = actionSets[i].getConfigurationElement()  
 .getDeclaringExtension();  
    reg.removeExtension(ext, new Object[] { actionSets[i] });  
}  
// Removing convert line delimiters menu.  
actionSetId = "org.eclipse.ui.edit.text.actionSet.convertLineDelimitersTo";   
for (int i = 0; i <actionSets.length; i++)  
{  
    if (!actionSets[i].getId().equals(actionSetId))  
        continue;  
    IExtension ext = actionSets[i].getConfigurationElement()  
            .getDeclaringExtension();  
   reg.removeExtension(ext, new Object[] { actionSets[i] });  
}

ATA100号规范可分为5个部分：  
　　5-12章为总体  
　　20-49为系统  
　　51-57为结构  
　　60-65为螺旋桨/旋翼  
　　70-91为发动机

|  |  |  |
| --- | --- | --- |
| 00 | INTRODUCTION | 介绍 |
| 00 | INTRODUCTION | 介绍 |
| 05 | TIME LIMITS / MAINTENANCE CHECKS | 维修时限和维护 |
| 06 | DIMENSIONS AND AREAS | 尺寸和图表 |
| 07 | LIFTING AND SHORING | 顶升 |
| 08 | LEVELING AND WEIGHING | 平衡和称重 |
| 09 | TOWING AND TAXIING | 牵引和滑行 |
| 10 | PARKING MOORING, STORAGE AND RETURN TO SERVICE | 停放和系留 |
| 11 | PLACARDS AND MARKINGS | 标牌和标志 |
| 12 | SERVICING | 勤务 |
| 20 | STANDARD PRACTICES | 系统部分标准施工 |
| 21 | AIR CONDITIONING | 空调 |
| 22 | AUTO FLIGHT | 自动飞行 |
| 23 | COMMUNICATIONS | 通讯 |
| 24 | ELECTRICAL POWER | 电源 |
| 25 | EQUIPMENT / FURNISHINGS | 设备及装饰 |
| 26 | FIRE PROTECTION | 防火 |
| 27 | FLIGHT CONTROLS | 飞行操纵 |
| 28 | FUEL | 燃油 |
| 29 | HYDRAULIC POWER | 液压 |
| 30 | ICE AND RAIN PROTECTION | 防冰防雨 |
| 31 | INDICATING / RECORDING SYSTEMS | 仪表（指示记录） |
| 32 | LANDING GEAR | 起落架 |
| 33 | LIGHTS | 灯光 |
| 34 | NAVIGATION | 导航 |
| 35 | OXYGEN | 氧气 |
| 36 | PNEUMATIC | 引气 |
| 37 | VACUUM | 真空 |
| 38 | WATER / WASTE | 水和排污 |
| 38 | AIRBORNE AUXILIARY POWER APU | 机载辅助动力装置 |
| 51 | STANDARD PRACTICES AND STRUCTURES - GENERAL | 结构部分标准施工 |
| 52 | DOORS | 门 |
| 53 | FUSELAGE | 机身 |
| 54 | NACELLES / PYLONS | 发动机短舱、吊架 |
| 55 | STABILIZERS | 安定面 |
| 56 | WINDOWS | 窗 |
| 57 | WINGS | 机翼 |
| 61 | PROPELLERS-General | 螺旋桨-概述 |
| 65 | ROTORS | 旋翼 |
| 70 | STANDARD PRACTICES - ENGINES | 动力装置部分标准施工 |
| 71 | POWER PLANT | 动力装置 |
| 72 | ENGINE | 发动机 |
| 73 | ENGINE FUEL AND CONTROL | 发动机燃油及调节 |
| 74 | IGNITION | 点火 |
| 75 | AIR | 空气 |
| 76 | ENGINE CONTROLS | 发动机控制 |
| 77 | ENGINE INDICATING | 发动机指示 |
| 78 | EXHAUST | 排气 |
| 79 | OIL | 滑油 |
| 80 | STARTING | 起动 |
| 81 | TURBINES | 涡轮 |
| 82 | WATER INJECTION | 喷水 |
| 83 | ACCESSORY GEAR BOXES | 附件齿轮箱 |
| 91 | CHARTS | 图表（WDM中） |
| 93 | PANELS | 面板 |