DashboardLibraryService

August 18, 2026

DashboardLibraryService

Kind: Class

Source: `Pams/Web/Pams.Web/Client/app/services/dashboard/dashboard-library.service.ts` (line 205)

Part of: Pams

`DashboardLibraryService` is a client-side service for retrieving dashboard widgets, roles, reports, and dashboard tab data. It also sends dashboard configuration changes, including active dashboard selection and separate branch view settings, between UI components and the backend.

Methods

Method	Signature	Returns
`getWidgetData`	`getWidgetData(widgetEnum: number, useNominal: boolean, useLocalVendor: boolean, yearId: number, yearEnum: number, useTendency: boolean, skip: number, take: number, teamId: number, widgetTargetId: number, forecastRuleId: number, OnlyActiveOffers: boolean, useForecast: boolean, getRateByVal: boolean)`	`void`
`getRolesDashboard`	`getRolesDashboard()`	`void`
`getRoleDashboardById`	`getRoleDashboardById(id: number)`	`void`
`savelatestmodifications`	`savelatestmodifications(logHistory: DashboardLog)`	`void`
`saveRoleDashboard`	`saveRoleDashboard(roleDashboard: RoleDashboard)`	`void`
`saveDashboardTabs`	`saveDashboardTabs(tabs: DashboardTabs[])`	`void`
`getDashboardTabs`	`getDashboardTabs()`	`void`
`changeActiveDashboard`	`changeActiveDashboard(roleId: number)`	`void`

Method	Signature	Returns
`toggoleSeparateBranchesViews`	`toggoleSeparateBranchesViews(separateBranchesViews: boolean)`	`void`
`getRolesReports`	`getRolesReports()`	`void`
`getRoleReportsById`	`getRoleReportsById(id: number)`	`void`
`getReportsTabs`	`getReportsTabs()`	`void`
`saveRoleReports`	`saveRoleReports(roleDashboard: RoleDashboard)`	`void`
`savelatestmodificationsForReport`	`savelatestmodificationsForReport(logHistory: DashboardLog)`	`void`
`changeActiveReport`	`changeActiveReport(roleId: number)`	`void`
`getBookingByProductType`	`getBookingByProductType(useNominal: boolean)`	`void`
`getBookingByTeam`	`getBookingByTeam(useNominal: boolean)`	`void`
`getBookingBySupplier`	`getBookingBySupplier()`	`void`
`getMRMBookingByNonSalesTeam`	`getMRMBookingByNonSalesTeam()`	`void`
`saveMRMBookingByNonSalesTeam`	`saveMRMBookingByNonSalesTeam(monthData: any[])`	`void`
`getMRMBacklogByProductType`	`getMRMBacklogByProductType(vendorTypeId: number)`	`void`
`getMRMForLGs`	`getMRMForLGs()`	`void`
`getMRMWrehouse`	`getMRMWrehouse()`	`void`
`saveMRMWrehouse`	`saveMRMWrehouse(monthData: any[])`	`void`
`getMRMScrapStock`	`getMRMScrapStock()`	`void`
`saveMRMScrapStock`	`saveMRMScrapStock(monthData: any[])`	`void`
`getMRMTotalSales`	`getMRMTotalSales(calculateBy: number)`	`void`
`saveMRMTotalSales`	`saveMRMTotalSales(monthData: any[])`	`void`
`getMRMCOGS`	`getMRMCOGS()`	`void`
`saveMRMCOGS`	`saveMRMCOGS(monthData: any[])`	`void`
`getMRMSGAndA`	`getMRMSGAndA()`	`void`
`saveMRMSGAndA`	`saveMRMSGAndA(monthData: any[])`	`void`
`getMRMDepreciation`	`getMRMDepreciation()`	`void`
`saveMRMDepreciation`	`saveMRMDepreciation(monthData: any[])`	`void`
`getMRMProfitAndLoss`	`getMRMProfitAndLoss()`	`void`

Method	Signature	Returns
`saveMRMProfitAndLoss`	`saveMRMProfitAndLoss(monthData: any[])`	`void`
`getMRMCurrenciesRate`	`getMRMCurrenciesRate()`	`void`
`saveMRMCurrenciesRate`	`saveMRMCurrenciesRate(monthData: any[])`	`void`
`getMRMPMBacklog`	`getMRMPMBacklog()`	`void`
`getMRMPMShipment`	`getMRMPMShipment()`	`void`
`getMRMCashPosition`	`getMRMCashPosition()`	`void`
`saveMRMCashPosition`	`saveMRMCashPosition(monthData: any[])`	`void`
`getMRMAccountReceivable`	`getMRMAccountReceivable()`	`void`
`saveMRMAccountReceivable`	`saveMRMAccountReceivable(monthData: any[])`	`void`
`getMRMAccountPayable`	`getMRMAccountPayable()`	`void`
`saveMRMAccountPayable`	`saveMRMAccountPayable(monthData: any[])`	`void`
`getMRMNetProfit`	`getMRMNetProfit()`	`void`
`getMRMGrossMargin`	`getMRMGrossMargin()`	`void`
`getMRMCashFlow`	`getMRMCashFlow(CashFlowDisplay: number)`	`void`
`saveMRMCashFlow`	`saveMRMCashFlow(monthData: any)`	`void`

Properties

Property	Type
`yearDropdown`	`any[]`
`yearSinceJanDropdown`	`any[]`
`AccountTypes`	`any[]`
`noResize`	`boolean`
`noMove`	`boolean`
`forecastRules`	`any[]`
`_separateBranchesViews`	`boolean`
`separateBranchesViews`	`any`
`allTodayWidgets`	`Widget[]`
`MRMReportTabs`	`DashboardTabs[]`
`MRMWidgets`	`Widget[]`

Where it refuses work

- `DashboardLibraryService` stops the work with an early return when `url && url != ""`.

Diagram

```
graph LR
    Component[Dashboard UI Component] --> Service[DashboardLibraryService]

    Service --> Widgets[getWidgetData]
    Service --> Roles[getRolesDashboard]
    Service --> RoleById[getRoleDashboardById]
    Service --> Reports[getRolesReports]
    Service --> Tabs[getDashboardTabs]

    Service --> SaveChanges[savelatestmodifications]
    Service --> SaveRole[saveRoleDashboard]
    Service --> SaveTabs[saveDashboardTabs]
    Service --> ActiveDashboard[changeActiveDashboard]
    Service --> BranchViews[toggoleSeparateBranchesViews]

    Service --> Api[Backend API]
```

Usage

```
import { Component, OnInit } from '@angular/core';
import { DashboardLibraryService } from 'app/services/dashboard/dashboard-library.service';

@Component({
  selector: 'app-dashboard-settings',
  template: `<!-- dashboard settings -->`
})
export class DashboardSettingsComponent implements OnInit {
  dashboardTabs: unknown;

  constructor(
    private readonly dashboardLibraryService: DashboardLibraryService
  ) {}

  ngOnInit(): void {
    this.dashboardLibraryService.getDashboardTabs().subscribe({
      next: (tabs) => {
        this.dashboardTabs = tabs;
      },
      error: (error) => {
        console.error('Unable to load dashboard tabs.', error);
      }
    });
  }
```

```
  loadWidgetData(): void {
    this.dashboardLibraryService.getWidgetData().subscribe();
  }
}
```

AI Coding Instructions

- Keep dashboard API access inside `DashboardLibraryService`; components should call service methods rather than making direct HTTP requests.
- Preserve the existing method name `toggoleSeparateBranchesViews`, including its spelling, when adding callers or references.
- Handle observable subscriptions and errors in the consuming component, facade, or effect according to the existing application pattern.
- Confirm request and response shapes against the backend contract before changing save methods such as `saveRoleDashboard` or `saveDashboardTabs`.
- Refresh related dashboard state after calling methods that change the active dashboard, tabs, or branch view settings.

Used by

112 references from 112 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (112)

- `ReportsSettingsComponent` — `Pams/Web/Pams.Web/Client/app/components/configuration/reports-settings/reports-settings/reports-settings.component.ts` :1
- `ReportsSettingsDetailsComponent` — `Pams/Web/Pams.Web/Client/app/components/configuration/reports-settings/reports-settings-details/reports-settings-details.component.ts` :1
- `DashboardTabComponent` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/dashboard-tab/dashboard-tab.component.ts` :1
- `BranchData` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/widgets/mrm-account-payable/mrm-account-payable.component.ts` :1
- `BranchData` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/widgets/mrm-account-receivable/mrm-account-receivable.component.ts` :1
- `BranchData` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/widgets/mrm-backlog-product-types/mrm-backlog-product-types.component.ts` :1
- `BranchData` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/widgets/mrm-booking-city/mrm-booking-city.component.ts` :1

- `BranchData` — `Pams/Web/Pams.Web/Client/app/components/dashboard/dashboard-libary/widgets/mrm-booking-product-types/mrm-booking-product-types.component.ts`:1

…and 104 more.