CRMLinqDataContext

August 17, 2026

CRMLinqDataContext

Kind: Class

Source: `Pams/Data/Pams.Data/CRMLinq.designer.cs` (line 26)

Part of: Pams

`CRMLinqDataContext` is a LINQ to SQL data context for CRM reference and task-related tables. It exposes typed `Table<T>` properties for activity types, participant types, topic templates, statuses, assignee access types, task assignee lists, and priorities so server-side code can query and update mapped records.

Diagram

```
graph LR
    App[Application Service] --> Context[CRMLinqDataContext]
    Context --> ActivityTypes[ActivityTypes]
    Context --> ActivityActionTypes[ActivityActionTypes]
    Context --> ActivityLinks[ActivityLinks]
    Context --> ParticipantTypes[ParticipantTypes]
    Context --> TopicTemplates[TopicTemplates]
    Context --> TopicTemplateItems[TopicTemplateItems]
    Context --> Status[Status]
    Context --> AssigneeAccessTypes[AssigneeAccessTypes]
    Context --> TaskAssigneeLists[TaskAssigneeLists]
    Context --> Priorities[Priorities]
```

Usage

```
type Status = {
  id: number;
  name: string;
};

async function loadStatuses(): Promise<Status[]> {
  const response = await fetch("/api/crm/statuses");
```

```
  if (!response.ok) {
    throw new Error("Unable to load CRM statuses.");
  }

  return response.json();
}

const statuses = await loadStatuses();
console.log(statuses);
```

AI Coding Instructions

- Create `CRMLinqDataContext` instances in server-side code where the database connection is available.
- Query mapped tables through their typed properties, such as `context.Status` or `context.Priorities`.
- Keep LINQ queries scoped to the required records before materializing results.
- Treat this generated designer file as mapping code; place custom query logic in separate partial classes or services.
- Expose data to TypeScript clients through API endpoints rather than sending the LINQ data context to the browser.

Used by

28 references from 28 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (28)

- `FilesController` — `Pams/API/Pams.API/Controllers/Common/FilesController.cs`:27
- `ActivitiesController` — `Pams/API/Pams.API/Controllers/Tasks/Activities/ActivitiesController.cs`:17
- `PriorityController` — `Pams/API/Pams.API/Controllers/Tasks/PrioritiesController.cs`:22
- `StatusController` — `Pams/API/Pams.API/Controllers/Tasks/StatusController.cs`:10
- `UnitOfWork` — `Pams/Logic/Pams.Business/Component/UnitOfWork.cs`:12
- `HasPermissionModel` — `Pams/Logic/Pams.Business/Component/Security/RolePermissionLogic.Logic.cs`:11
- `Comment` — `Pams/Logic/Pams.Business/Entity/Common/Comments.Model.cs`:16
- `Tag` — `Pams/Logic/Pams.Business/Entity/Common/Tags.Model.cs`:16

…and 20 more.