CNAPI.GetNodeTypes

نحوه فراخوانی این API با چیزی که در حال حاضر پیاده سازی کردی فرقی نداره. فقط توی خروجی API داخل شیء هر کدوم از NodeType ها یه ویژگی IsCategory هم اضافه شده که اگر مقدارش true باشه، به این معنیه که اون آیتم یه «**دسته**» ست و اگر مقدار نداشته باشه یا false باشه، اون آیتم یه کلاسه. توی نسخه سازمانی همیشه همه آیتما کلاس هستن.

موقع صدا زدن این API یه پارامتر دیگه هم میتونی ارسال کنی به اسم Archive که اگر مقدارش true باشه، خروجی شامل کلاس های آرشیو شده یا همون حذف شده خواهد بود. این مورد رو امیرحسین هنوز توی طراحیش توجه نکرده اما لازمه که اضافه کنه؛ چون این امکان رو داریم که کلاس های حذف شده رو برگردونیم.

نکات:

۱. دسته ها همیشه root درخت هستن و نمیتونن زیرشاخه دسته یا کلاس دیگه ای باشن.
۲. توی نسخه SaaS:
   a. کلاس ها نمیتونن زیر کلاس داشته باشن.
   b. یک کلاس نمیتونه زیرشاخه کلاس دیگه ای باشه و فقط میتونه زیرشاخه یه دسته باشه.
   c. یک کلاس میتونه از یه دسته به دسته دیگه ای منتقل بشه.
   d. یک کلاس میتونه زیر هیچ دسته ای نباشه و خودش root باشه.
   e. کلاس ها آیکون دارن؛ اما دسته ها آیکون ندارن.
   f. همیشه کنار اسم دسته ها یه آیکون مثلثی شکل که نشونه داشتن زیر شاخه هست وجود داره؛ حتی اگر اون دسته هیچ کلاسی نداشته باشه.
   g. در صورتی که تعدادی کلاس و تعدادی دسته داشته باشی که هم زمان root هستن، این امکان وجود داره که همه رو با هم مرتب کنی؛ یعنی ممکنه یک در میون یه کلاس و یه دسته قرار بگیرن.
۳. توی نسخه سازمانی:
   a. دسته نداریم و همه آیتما کلاس هستن.
   b. کلاس ها میتونن زیر کلاس داشته باشن و زیر کلاس ها هم میتونن زیر کلاس داشته باشن و عمق درخت نامحدوده.
   c. کنار کلاسایی که زیرکلاس ندارن آیکون مثلثی شکل که به معنای داشتن زیر کلاسه وجود نداره.
   d. کلاس ها میتونن به صورت drag & drop مرتب بشن یا از زیرشاخه یک کلاس به زیرشاخه یک کلاس دیگه منتقل بشن یا حتی تبدیل به root بشن.

CNAPI.AddNodeType({ Name: base64_string, ParentID: “”, IsCategory: bool })

در صورتی که داری دسته ایجاد می کنی، مقدار IsCategory باید true باشه. در صورتی که داری یک کلاس رو به عنوان زیرشاخه یه دسته یا یه کلاس دیگه اضافه می کنی، باید مقدار ParentID برابر با مقدار NodeTypeID اون دسته یا کلاس باشه.

CNAPI.RenameNodeType({ NodeTypeID: “”, Name: base64_string })

برای تغییر نام یک کلاس یا دسته باید از این API استفاده کنی.

CNAPI.MoveNodeType({ NodeTypeID: “”, ParentID: “” })

برای اینکه پدر یک کلاس رو تغییر بدی باید از این API استفاده کنی. در صورتی که بخوای کلاس رو به root ببری، باید مقدار ParentID برابر با null باشه.

CNAPI.RemoveNodeType({ NodeTypeID: “”, RemoveHierarchy: bool })

برای حذف کردن یک کلاس یا دسته باید از این API استفاده کنی. در صورتی که مقدار پارامتر RemoveHierarchy برابر با true باشه همزمان با دسته یا کلاسی که داری حذفش میکنی، همه زیرشاخه هاشم حذف میکنه. در صورتی هم که برابر با false باشه؛ همه زیرشاخه ها رو root می کنه.

CNAPI.SetNodeTypesOrder({ NodeTypeIDs: “id1|id2|id3|...” })

برای مرتب سازی لیستی از کلاس ها یا دسته ها از این API باید استفاده کنی. فقط یه پارامتر میگیره به اسم NodeTypeIDs که یه string هست و باید مقادیر NodeTypeID کلاس هایی که میخوای مرتب کنی رو به ترتیب و با کاراکتر جداکننده ’|‘ (اون خطه که عمودیه) ارسال کنی. یعنی:

[id1, id2].join(‘|’)

CNAPI.RecoverNodeType({ NodeTypeID: “” })

در صورتی که بخوای یه کلاس رو که حذف شده برگردونی، باید از این API استفاده کنی. از این API برای برگردوندن هم دسته ها و هم کلاس ها میتونی استفاده کنی؛ اما به نظر میرسه که نیازی نباشه که اصلا دسته ها رو بشه برگردوند.