بسم الله الرحمن الرحیم

نام و نام خانوادگی دانشجو : محمد مهدی عبداللهی

شماره دانشجویی : ۰۱۲۲۱۰۳۳۷۲۰۰۲۵

عنوان تحقیق : تفاوت و انواع متغیر ها ، انواع حافظه در رم و reference type – value type

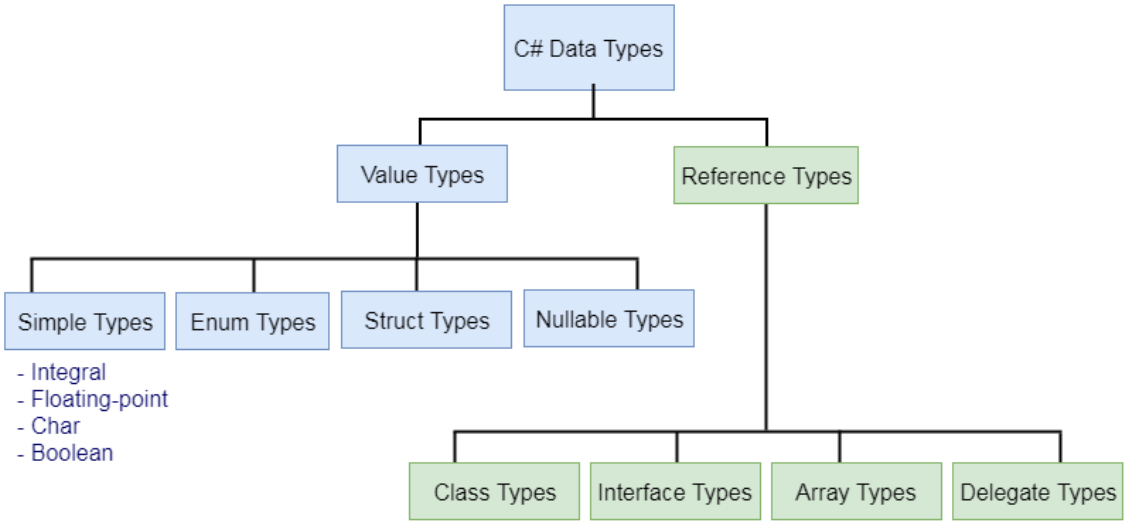
نام استاد : مهندس یاریان

نام درس : برنامه نویسی سمت سرور

انواع Data type

زبان سی شارپ یکی از زبان‌های سطح بالا است که در دسته زبان های نوع‌دهی نیرومند قرار دارد. زبان نوع‌دهی نیرومند به این معنی است که باید نوع متغیر را مشخص کنیم و با این کار مقادیر مجاز برای یک متغیر تعیین می شود.

به طور کلی نوع داده در سی شارپ به دو دسته تقسیم می شود. این دو دسته عبارتند از داده های مشخص و داده های ارجاعی. داده های مشخص عبارتند از داده های ساده ای مثلint ، Float، boolوchar ، داده های enum، struct، داده های .NULL داده های ارجاعی نیز شامل داده هایClass ، Interface، Delegateو arrayمی‌باشد. .



انواع Data type در سی شارپ بر اساس دسته بندی

داده های مشخص

زمانی که یک مقدار به طور مستقیم در یک داده ذخیره شود به آن داده مشخص گفته می‌شود.

نوع داده‌ها قبلا در کتابخانه‌های سی شارپ تعیین شده‌اند. در اینجا به بررسی برخی از انواع Data type ها می‌پردازیم:

Byte

این نوع داده اعداد صحیح بین ۰ تا ۲۵۵ را در خود ذخیره کرده و ۸ بیت فضا در حافظه را اشغال می‌کند.

Sbyte

این نوع داده می‌تواند اعداد بین -۱۲۸ تا +۱۲۸ را در ۸ بیت حافظه ذخیره کند.

Short

این نوع داده اعداد صحیح بین -۳۲۷۶۸ تا +۳۲۷۶۸ را در خود ذخیره کرده و ۱۶ بیت فضا در حافظه را اشغال می‌کند.

Int

این نوع داده اعداد صحیح بین -۲۱۴۷۴۸۳۶۴۸ تا +۲۱۴۷۴۸۳۶۴۸ را در خود ذخیره کرده و ۳۲ بیت فضا در حافظه را اشغال می‌کند.

Uint

این نوع داده اعداد صحیح بین ۰ تا +۴۲۹۴۹۶۷۲۹۵ را در خود ذخیره کرده و ۳۲ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند u استفاده شود.

Long

این نوع داده اعداد صحیح بین -۹۲۲۳۳۷۲۰۳۶۸۵۴ تا +۹۲۲۳۳۷۲۰۳۶۸۵۴ را در خود ذخیره کرده و ۶۴ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند l استفاده شود. برای نمایش اعداد بزرگ در سی شارپ از این نوع داده استفاده می شود.

Ulong

این نوع داده اعداد صحیح بین ۰ تا +۱۸۴۴۶۷۴۴۰۷۳۷۰۹۵۵۱۶۱۵ را در خود ذخیره کرده و ۶۴ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند ul استفاده شود.

Float

این نوع داده اعداد اعشاری بین – ۳.۴۰۲۸۲۳ e۳۸تا +۳.۴۰۲۸۲۳ e۳۸را در خود ذخیره کرده و ۳۲ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند f استفاده شود.

Double

این نوع داده اعداد اعشاری بین -۱.۷۹۷۶۹۳۱۳۴۸۶۲۳۲ e۳۰۸تا +۱.۷۹۷۶۹۳۱۳۴۸۶۲۳۲ e۳۰۸را در خود ذخیره کرده و ۶۴ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند d استفاده شود

تفاوت بین Float و Double در این است که Float تا ۷ رقم اعشاری را می تواند در خود ذخیره کند اما Doubleمی تواند ۱۴ الی ۱۵ رقم اعشاری را در خود ذخیره کند.

Decimal

یکی دیگر از روش های نمایش داده ها با نوع اعشاری درC# ، استفاده از Decimal است. این نوع داده اعداد اعشاری در بازه مثبت و منفی ۱.۰ x ۱۰e-۲۸ تا مثبت و منفی ۷.۹ x ۱۰e۲۸ را در خود ذخیره کرده و ۱۲۸ بیت فضا در حافظه را اشغال می کند. در هنگام کد نویسی نیز باید از پسوند m استفاده شود.
این نوع داده بیشتر برای کارهای حسابداری مورد استفاده قرار می گیرد و می تواند ۲۸ الی ۲۹ رقم اعشار دقت داشته باشد. اما نسبت به Double و Float رنج کمتری دارد.
نکته: حرف e یا E زمانی که با نوع داده اعشاری همراه شده باشند نشان دهنده به توان ۱۰ می باشد.

حافظهstack

در بخش user-space حافظه قرار دارد و به صورت خودکار توسط CPU مدیریت می شود. متغیرهای غیر استاتیک، پارامتر های ارسالی به توابع و آدرس های مربوط به return توابع در این حافظه ذخیره می شوند. اندازه حافظه stack ثابت است به همین دلیل به آن static memory گفته می شود.
در این حافظه اطلاعات پشت سر هم و به ترتیب قرار می گیرند به این صورت که آخرین داده ذخیره شده در بالای stackقرار می گیرد و به اصطلاح push می شود، حال اگر قصد برداشتن اطلاعات یا به اصطلاح pop کردن اطلاعات را داشته باشیم آخرین اطلاعات وارد شده در stack را در اختیار داریم. به این الگوریتم LIFO(Last In First Out)می گویند. مثال پر کاربرد در توضیح stack خشاب اسلحه (آخرین گلوله ای که در خشاب قرار داده می شود اولین گلوله ای است که شلیک می شود) و یا بشقاب های روی هم چیده شده (آخرین بشقابی که روی سایر بشقاب ها قرار داده می شود اولین بشقابی است که برداشته می شود) است.
از آنجا که در حافظه stack نیازی به پیدا کردن فضای خالی در حافظه نیست و محل قرارگیری اطلاعات مشخص است (بالای حافظه) بنابراین این حافظه سریع تر از حافظه heap است.
پارامتر ها و اطلاعات مربوط به توابع برای اجرا و کنترل آن ها در این حافظه ذخیره می شوند. تابعی که در بالای stackقرار دارد تابعی است که در حال اجراست و بعد از اتمام کار تابع یا بروز خطا در اجرای تابع، حافظه

اختصاص داده شده به تابع از stack حذف می شود و حافظه اشغال شده آزاد می شود. زمانی که یک thread تعریف می شود در stack قرار می گیرد.
خطایی که ممکن است در اثر استفاده نادرست از حافظه stack رخ دهد stack overflow است. از جمله دلایل stack overflow یا سر ریز می توان به استفاده از متغیرهای محلی حجیم که منجر به کاهش فضای آزاد در stack و تخریب یا corrupt شدن بخشی از memory اشاره کرد.

حافظه Heap

حافظه Heap در قست user-space حافظه مجازی قرار دارد و به صورت دستی توسط برنامه نویس مدیریت می شود .Heap مربوط به زمان اجرا (runtime) است و فضای اشغال شده در heap با اتمام کار تابع آزاد نمی شوند و تا زمانی که Garbage Collector این فضا را آزاد کند یا توسط برنامه نویس داده ها از حافظه heap پاک نشوند در این فضا باقی می ماند. اندازه حافظه heap متغیر است به همین دلیل به آن dynamic memory گفته می شود.
در این نوع از حافظه برای ذخیره مقادیر ابتدا محاسبه ای توسط سیستم عامل صورت می گیرد تا اولین فضای حافظه ای که اندازه آن متناسب با اندازه ای که مورد نیاز ماست را پیدا کند، در صورت وجود این میزان از حافظه درخواستی آن را به صورت رزرو شده درمی آورد تا بقیه برنامه ها به این فضا دسترسی نداشته باشند، سپس آدرس ابتدای این فضای محاسبه شده به صورت یک اشاره گر (pointer) در اختیارمان قرار می دهد (یا به اصلاح allocating).
متغیر ها به صورت پیش فرض در این حافظه قرار نمی گیرند و اگر قصد ذخیره متغیر ها در این حافظه را داشته باشیم باید به صورت دستی این اقدام انجام شود. متغیر هایی که در heap ذخیره می شوند به طور خودکار حذف نمی شوند و باید توسط برنامه نویس و به صورت دستی حذف شوند. به طور کلی مدیریت حافظه heap به صورت دستی توسط برنامه نویس انجام می شود. آرایه های داینامیک در heap ذخیره می شوند.
در صورتی که داده های ما از تعدا block های پشت سر هم در حافظه بیشتر باشد یا در صورت تغییر حجم داده ها در زمان های مختلف (تغییر سایز داده ها امکان پذیر است)، سیستم عامل داده ها را به صورت تکه تکه در block های حافظه ذخیره خواهد کرد.
به دلیل محاسبات برای یافتن آدرس شروع حافظه و در اختیار گرفتن pointer حافظه heap نسبت به stack کندتر است. همچنین اگر داده ها به صورت پشت سر هم در block های حافظه قرار نگرفته باشند (این احتمال بسیار زیاد است) موجب کندی در بازیابی اصلاعات خواهد شد.

وقتی که نمونه‌ای از یک کلاس ایجاد می‌کنیم این مقدار در Heap ذخیره می‌شود و وقتی که کار آن به پایان می رسد garbage collector حافظه را آزاد می کند و اگر موفق به این کار نشود، برنامه نویس باید به صورت دستی حافظهheap را آزاد کند، در غیر این صورت Memory leak اتفاق می افتد که به معنی in use نگه داشتن فضای حافظه برای اشیایی است که دیگر از آن ها در برنامه استفاده نمی شود و garbage collector قادر به آزاد سازی فضایی که آن ها اشغال کرده اند نیست.

به طور کلی Value Type ها (primitive type) فضای زیادی اشغال نمی کنند و در stack ذخیره می شوند. برای دسترسی به متغیر هایValue Type ، مقدار آن به صورت مستقیم از حافظه stack خوانده می شود، مثلا زمانی که متغیری تعریف می کنیم آن متغیر به همراه مقدار آن در stack قرار می گیرد.
برای دسترسی به متغیرهایReference Type ، ابتدا با مراجعه Stack و دریافت آدرس متغیر در Heap به شئ مربوط به متغیر دسترسی خواهیم داشت Reference Type .ها در حافظه heap نگهداری می شوند. زمانی که یک شی از کلاس ایجاد می کنیم ابتدا متغیری که شی به آن assign شده است با مقدار null در حافظه stack قرار می گیرد، سپس شی در heap ذخیره شده و پس از ذخیره سازی در heap آدرس شی در stack جایگزین nullمی شود.
همچنین reference type ها به dynamic memory و value type ها به static memory نیاز دارند.
در صورت نیاز بهdynamic memory ، باید امکان دسترسی به heap فراهم باشد و اگر نیازمند static memoryباشیم، stackمحل ذخیره سازی خواهد بود.

اما تفاوت value type و reference type و طراحان زبان برنامه نویسی: زبان برنامه نویسی سی شارپ برای افزایش پرفورمنس و سرعت زبان سی شارپ داده های از نوع اصلی رو که معمولا بیشتر باهاش سر و کار داریم و نیاز هست خیلی سریع اجرا بشن و همچنین ظرفیت محدود و مشخصی دارن رو در بخش استک حافظه ی رم ذخیره میکنن. انواع داده ای مانند bool,int , double, struct و...
و نوع های دیگه مثل enum و class رو در حافظه ی heap ذخیره و استفاده میکنند.

خب وقتی میگیم value type یعنی متغیر ما از حافظه ی استک استفاده میکنه.

شیوه ذخیره سازی و فراخوانی مقادیر در استک و هیپ متفاوته و بحث مفصلی میطلبه اما فقط اشاره هایی مختصر بهشون میکنیم.

در استک مقدار داده مستقیما در خونه حافظه ذخیره میشه, مثلا:

int me =۲

در این مثال مقدار عددی متغیر me که ۲ هست در یکی از خونه های حافظه در بخش استک ذخیره شده و اسم اون خونه me هست.

اما وقتی میگیم نوع داده reference type هست یعنی از حافظه ی هیپ استفاده میشه. به عبارت دیگر آدرس خونه حافظه رو در متغیر داریم و نه مقدارش.

فرض کنید متغیری داریم از نوع reference type به اسم var که مقدار "hello" رو میخوایم درش ذخیره کنیم. اتفاقی که در عمل میفته اینه:

#با توجه به اینکه هر کاراکتر ۱ بایت فضا اشغال میکنه و کلمه hello دارای ۵ کاراکتر هست کلا ۵ بایت فضا نیاز داره پس خونه ۱۰۰۰ رو تا ۱۰۰۵ رو مقدار دهی کن بدین شکل که در هر خونه یک کاراکتر ، مثلا خونه ۱۰۰۰ h خونه ۱۰۰۱ e تا خونه ۱۰۰۵

و یک خونه دیگه هم برای ذخیره آدرس استفاده میشه.

همچنین اگر مقداری براش تعیین نشه اصطلاحا null یا پوچ خواهد بود که این null بودن رو در value typeها نداریم.

به طور خلاصه تفاوت value type و reference type رو میشه اینطور بیان کرد که:

۱-داده های نظیر int float bool و value type ... هستند.

داده هایی نظیرenum و class از نوع reference هستند و struct از نوع value هستند

۲-در داده های از نوع value type مقدار داده به طور مستقیم در حافظه ذخیره میشود.

درreference type ها آدرس آن درون حافظه ذخیره میشه و مقدار آن توسط آدرس به طور غیر مستقیم قابل دسترسی هست.

۳-تغییر و انجام هر عملیات روی یک متغیر از نوع reference باعث تغییر سایر متغیرهایی هم میشود که به همان خانه حافظه اشاره میکنند.

اما در value اینگونه نیست.