دنیای صفر و یک

شماره نهم - خرداد ۹۹

دانشگاه صادر کننده مجوز : دانشگاه تهران

صاحب امتیاز :

انجمن علمی مهندسی کامپیوتر دانشگاه تهران، پردیس فارابی

مدیر مسئول :

علی داودآبادی فراهانی

سردبیر :

محمد خشنود

هیأت تحریریه :

پیام رنجبر، محمد رشیدخان، عبدالحمید ناظرپناهی،
سید علی آل طه، سید محمد مهدی هاشمی، امیرعلی گرجی

صفحه آرا :

سید محمد مهدی هاشمی

نویسندگان این شماره :

سید علی آل طه، امیرعلی گرجی، عبدالحمید ناظرپناهی، پیام رنجبر

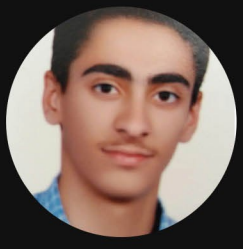
راه‌های ارتباطی :

Email :
Hashemi.smm79@yahoo.com
Dscorpion78@gmail.com

Telegram:
@CallTech
@Mhdr78

1

یادگیری تقویتی / Reinforcement Learning



سید علی آل‌طه

یکی از شاخه های Machine Learning که مبتنی بر یادگیری بر اساس بازخوردهای عمل است یادگیری تقویتی نام دارد. یادگیری ماشین و یا (ML) را میتوان اینگونه تعریف کرد: قادر ساختن کامپیوترها (به معنای عام) برای انجام استخراج و آنالیز داده ها و در نهایت استنتاج از آن‌ها.

از گذشته بشر از اطلاعات بدست آمده از اطرافش برای تصمیم گیری های کارآمد در زمینه‌های گوناگون استفاده میکرده است اما با پیچیده شدن دنیای اطلاعات روزانه حجم وسیعی از اطلاعات تولید می‌شود که تحلیل آن‌ها از توان بشر خارج است.اینجا جاییست که ML به کمک انسان می آید و زمینه‌ای را فراهم میکند تا مجبور نباشیم برای هر کاری که میخواهیم انجام دهیم کد دقیقی متناسب با آن بنویسیم بلکه اطلاعات را به رایانه بدهیم و او نتایج تحلیل را به ما نشان دهد و در این راه از الگوریتم های کم و بیش پیچیده و مبانی آماری و سایر ابزارهای ریاضیاتی استفاده میکند.

Reinforcement Learning یا همان یادگیری تقویتی کمی فراتر میرود چرا که در این نوع یادگیری برخلاف انواع دیگر ML از تکنیک های پیشرفته آماری برای تحلیل اطلاعات استفاده نمیشود بلکه عامل با انجام اعمال و دریافت بازخورد به معنای واقعی یاد میگیرد شاید به همین دلیل باشد که برخی RL را ترکیبی از یادگیری ماشین و هوش مصنوعی میدانند (که البته خود ML هم از AI مشتق شده است) در ادامه با جزئیات بیشتری فرآیند این یادگیری را مورد بررسی قرار میدهیم.

فرآیند یادگیری تقویتی

هنگام طراحی یک عامل طبیعتا محدودیت هایی چه بطور خواسته (طراح) و چه بطور ناخواسته تعیین میکنند که عامل چه کارهایی بتواند انجام دهد پس اعمال ممکن برای عاملی که RL بر روی آن پیاده سازی شده است مشخص است اما اطلاعات اولیه آن از محیط بسیار محدود یا صفر است.سپس عامل به انجام کنش در محیط میپردازد و بازخوردهایی از محیط میگیرد که این بازخوردها اطلاعات عامل در مورد محیط را افزایش میدهد؛ در نتیجه عامل در دفعات بعدی با اطلاعات و در نتیجه با کیفیت بیشتری اقدام به اتخاذ تصمیم ها میکند.این شیوه بسیار شبیه شیوه ایست که انسان از آن یاد میگیرد.آزمون و خطای اعمال در محیط ها به انسان کمک میکند درک خود را از محیط افزایش داده و به اتخاذ تصمیم های بهتر در مورد اعمال خود برسند.

به همین جهت است که گفته میشود عامل RL به معنای واقعی یاد میگیرد عاملی که از رویکرد RL بهره میبرد بیش از عاملی که به سایر رویکردها و شاخه های یادگیری ماشین مجهز شده به انسان شباهت دارد.برای روشن تر شدن این مطالب مثالی را در نظر بگیرید که در آن یک محیط شامل تعدادی خانه که در بعضی از آنها چاه وجود دارد توسط طراح مساله (که در هوش مصنوعی به آن دانای کل میگویند) طراحی شده است. اعمالی که

عامل میتواند انجام دهد حرکت در ۴ جهت اصلی است اما اطلاعاتی که در اختیار عامل قرار نمیگیرد این است که کدام خانه ها چاه دارند. سپس عامل در یک وضعیت اولیه در محیط مذکور رها میشود و به سیر و کشف محیط میپردازد.هرگاه عامل به چاه افتاد این یک بازخورد منفی برای او به حساب می آید ولی نکته اینجاست که او یاد میگیرد دیگر وارد آن خانه نشود و هرگاه وارد خانه ای شود و به چاه نیفتد می آموزد که وارد شدن به این خانه او را دچار مشکل نمیکند و بدین ترتیب رفته رفته کل محیط را میشناسد و راه را از چاه تشخیص میدهد.البته این مثال بسیار اولیه است و مشکلاتی که در دنیای واقعی بر سر راهمان قرار میگیرد را درنظر نمیگیرد؛ برای مثال چاه ها همیشه ثابت هستند و به بیانی با یک محیط کاملا ایستا و بدون تغییر سروکار داریم اما برای تبیین مفاهیم در ابتدای راه مفید است.

رویکردها و روش ها

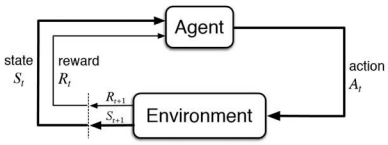
واقعیت اینست که روشهای مختلفی در یادگیری تقویتی بوجود آمده است که حاصل رویکردهای متفاوت سنجش اعمال عامل است.در محیط های کمی پیچیده تر از مثال قبل، فرمول بندی مساله اقتضا میکند که تابعی با عنوان ارزش گذاری طراحی شود که تعیین کند که نتیجه چه اعمالی پاداش و نتیجه کدام تنبیه است.اختلاف مذکور در رویکردها تعیین میکند که تابع ارزش گذاری چگونه باشد.نمونه ای از این اختلاف را میتوان اینگونه بیان کرد که برخی توابع نظر به سودهای نسبتا کوتاه مدت دارند در نتیجه هنگامی چنین تابعی تعیین کننده پاداش و جزا باشد عامل به عاملی تبدیل میشود که در هر مرحله بهترین تصمیم را میگیرد و به اصطلاح هوش مصنوعی یک عامل با رویکرد حریصانه خواهیم داشت.اما رویکرد دیگری که میتواند اتخاذ شود اینست که تابع به دنبال سودهای بلندمدت تر باشد و اگر احتمال بالایی بدهد که مثلا در انتهای راهی که میرود سود هنگفتی در انتظارش است حتی ضررهایی که بر سر راهش قرار میگیرند را میپذیرد تا در نهایت به آن سود نهایی برسد. میبینید که عامل مبتنی بر RL به بیان هوش مصنوعی، واقعا "خود مختار" است و عملکردش شباهت بسیاری به انسان-که غایت هوشمندی است-دارد.

یکی دیگر از مسائل مهمی که در فرمول بندی مساله و طراحی یک عامل مبتنی بر RL موثر است موضوع مستمر یا غیر مستمر بودن وظایف است.در نظر بگیرید که عاملی در پی یادگرفتن بازی شطرنج است.هر دور بازی که تمام میشود فارغ از برد یا باخت عامل و آموخته های او، میتوان در نظر گرفت که وظیفه او به پایان رسیده تا هنگامیکه دور بعدی بازی شروع شود، به این نوع از انجام وظیفه، غیر مستمر میگویند؛ در مقابل عاملی که سهام بورس را تحلیل و بر اساس آن تصمیم به خرید یا فروختن سهام میکند در یک اقیانوس از معاملات تمام نشدنی است لذا مجهز بودن چنین عاملی به آینده نگری به جای رویکرد حریصانه به توفیق هرچه بیشتر آن کمک شایانی خواهد کرد.

عامل مبتنی بر RL به بیان هوش مصنوعی، واقعا "خود مختار" است و عملکردش شباهت بسیاری به انسان که غایت هوشمندی است دارد.

یادگیری Q

یادگیری Q یکی از ساده ترین و در عین حال مفید مخصوصا برای علاقمندان تازه وارد است. Q در ابتدای عبارت، نماینده کلمه quality به معنای کیفیت است. در این نوع یادگیری جدولی طراحی میشود و عامل در هر وضعیتی که قرار میگیرد ابتدا اعمال ممکن را درنظر سپس یکی از آن اعمال را انجام میدهد و بازخورد را دریافت میکند و به کمک آن بازخورد جدول مذکور پر میشود. اینگونه عامل پس از مدت کوتاهی به بهترین عمل(عملی که موجب بیشترین سود میشود) در هر وضعیت پی میبرد.



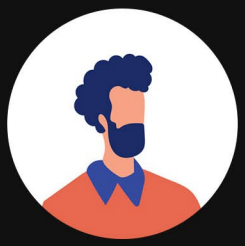
میبینید که یادگیری Q با رویکرد حریصانه عمل میکند و برای محیط های قطعی (مانند آنچه در مثال قسمت دوم دیدیم) بسیار مفید است.اما اشکالی که به این نوع از یادگیری وارد است اینست که برای محیط های غیر قطعی و پویا که دائما در حال تغییر وضعیت اند کاربردی ندارد؛ از آن گذشته در یک محیط بزرگ طراحی جدولی که اطلاعات مربوط به هر عمل درهر وضعیت را نشان دهد باعث سنگین شدن عامل میشود.

یک گام فراتر

ترکیب هوشمندی در عوامل RL و اتکا به محاسبات پیچیده و پرحجم در شبکه های عصبی مصنوعی باب تازه ای با نام Deep Reinforcement Learning را باز کرد.این نوع یادگیری کمک های بسیاری به انجام وظایف پیچیده تر از جمله انجام بازی هایی با محیط پویا کرد. پیشرفتهای چند سال اخیر در این زمینه بسیار پیچیده و بسیار داغ نوید بخش روزهای بهتری در علوم هوش مصنوعی است.

2

تکنولوژی RAID چیست؟



عبدالحمید ناظرپناهی

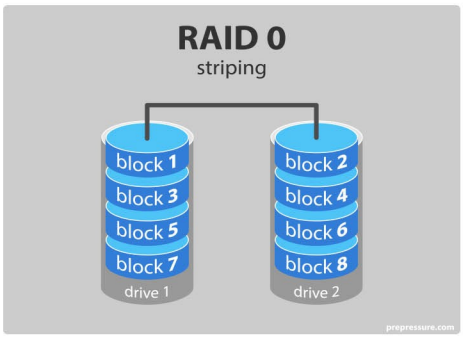
در دهه هشتاد میلادی سه دانشمند در عرصه کامپیوتر ایده ای را مطرح کردند که امروزه ما آن را به عنوان رِید (raid) میشناسیم. ایده این بود که با آرایه ای از دیسک های ارزان قیمت می توان به کارایی دیسک های گران قیمت تر دست یافت ضمن اینکه قابلیت اطمینان چند دیسک بیشتر از یک دیسک است.

اجـازه دهیـد تـا بـا یـک مثـال ایـده را بـه شـما معرفـی کنـم؛ فـرض کنیـد شـما مدیـر شـبکه یـک سـازمان اطلاعاتـی هسـتید. در ایـن شـبکه رکـن اصلـی اطلاعـات اسـت و بـه هیـچ عنـوان نبایـد اطلاعـات از دسـت بـرود. یـک خطـر جـدی بـرای اطلاعـات ، سـوختن و از دسـت رفتـن دیسـک هایـی اسـت کـه شـما اطلاعـات را روی آن ذخیـره کـرده ایـد. راه حلـی کـه ممکـن اسـت بـه نظـر برسـد گرفتـن نسـخه پشـتیبان از اطلاعـات اسـت امـا فـرض کنیـد حجـم اطلاعـات و سـرعت افزایـش حجـم اطلاعـات بـه گونـه ای اسـت کـه نسـخه پشـتیبان بـه تنهایـی امنیـت اطلاعـات را تضمیـن نمیکنـد. راه حـل چیسـت ؟ در ایـن مقالـه بـه بررسـی یـک راه حـل بـرای حـل ایـن مشکل می پردازیم.

RAID یـا redundant array of independent disks در واقـع آرایـه ای از چنـد دیسـک سـخت (HDD) مـی باشـد کـه بـه منظـور دسـتیابی بـه کارایـی ، پایایـی و گنجایـش بیشـتر بـا یکدیگـر اسـتفاده میشـوند. همچنیـن کل ایـن آرایـه بـرای سیسـتم عامـل میزبـان، بـه گونـه ای یکپارچـه رفتـار میکنـد مثـلا سیسـتم عامـل ویندوز تمامـی دیسـک هـا را بـه عنـوان یـک دیسـک شناسـایی مـی کنـد. البتـه توجـه داشـته باشـید کـه بـا اسـتفاده از ایـن تکنولـوژی باعـث میشـود تـا بخشـی از فضـای ذخیـره سـازی شـما دیگـر قابـل دسترسـی نباشـد امـا بـا توجـه بـه تضمینـی کـه بـرای اطلاعـات وجـود دارد ارزشـمندی آن مشـخص میشـود. اسـتفاده از ایـن تکنولـوژی باعـث افزایـش تـوان کارکـرد و افزایـش تحمـل خطـا مـی باشـد. اگرچـه از ایـن تکنولـوژی معمـولا در سـرور هـا اسـتفاده مـی شـود امـا امـروزه بعضـی از سیسـتم هـای خانگـی نیـز امـکان اسـتفاده از ایـن تکنولـوژی را بـه شـما مـی دهنـد. بـرای پیـاده سـازی آن بـه صـورت سـخت افـزاری نیـاز بـه مادربـرد و چنـد دیسـک سـخت داریـم و بـه صـورت نـرم افـزاری نیـاز بـه RAID Controller داریـم. چندیـن نـوع رِیـد داریـم کـه مختصـرا در زیـر بـه توضیـح سـه مـورد از مهم ترین آنها می پردازیم.

RAID0 / Strip

بـرای پیـاده سـازی ایـن نـوع از رِیـد نیـاز بـه حداقـل دو دیسـک داریـم. عملکـرد آن بـه ایـن صـورت اسـت کـه هنـگام نوشـتن اطلاعـات بـر روی دیسـک هـا نیمـی از اطلاعـات را روی دیسـک صفـر و نیمـی دیگـر از اطلاعـات را روی دیسـک یک ذخیره میکند.

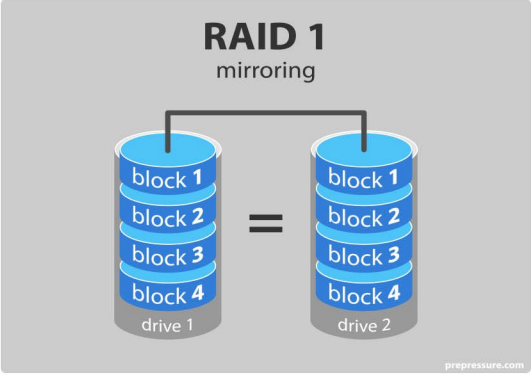


واضـح اسـت کـه اگـر یکـی از دیسـک هـا دچـار مشـکل شـوند قابلیـت بازگردانـی اطلاعـات وجـود نـدارد. بـه عبـارت دیگـر RAID0 تحمـل خطـا (fault-tolerance) نـدارد. مزیـت آن پیـاده سـازی آسـان و سـرعت خوانـدن و نوشـتن بـالای آن مـی باشـد. خوانـدن و نوشـتن در ایـن نـوع بـه صـورت

نوع به صورت همزمان می باشد.

RAID1 / Mirror

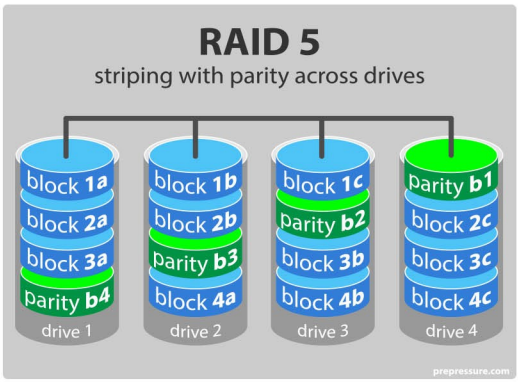
برای پیاده سازی آن به حداقل دو دیسک نیاز داریم. این نوع از رِید اطلاعات را به طور کامل در یک دیسک ذخیره میکند، سپس همان اطلاعات ذخیره شده را در دیسک بعدی نیز کپی میکند (مشابه آینه).



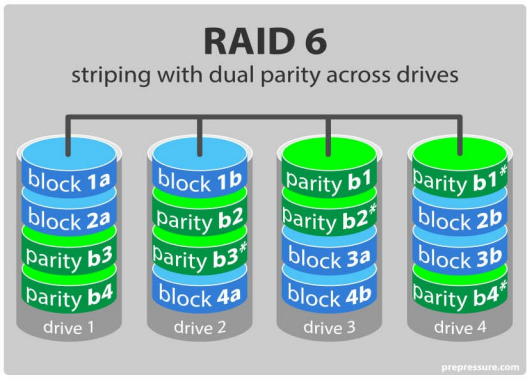
در این نوع از رِید تحمل خطا داریم، به این معنا که اگر یک دیسک دچار مشکل شود اطلاعات را از دیسک های دیگر می توان استخراج نمود . در این نوع از رِید سرعت نوشتن پایین است و سرعت خواندن بستگی به دیسک دارد. این نوع از رِید هزینه بر است و کمتر مورد استفاده قرار میگیرد. انواع مختلفی از رِید بعدها به وجود آمدند که پایه آنها همین نوع از رِید می باشد .

RAID5 / Parity

برای انجام این نوع از رِید نیاز به حداقل ۳ دیسک داریم که یک دیسک نقش Parity یا مکمل را ایفا میکند و دو دیسک دیگر برای ذخیره سازی اطلاعات می باشند. دیسک مکمل در واقع شامل بعضی از متادیتاها (meta-data) و بیت های مکمل می باشد. تا حداکثر یک دیسک در این نوع رِید، تحمل خطا وجود دارد.



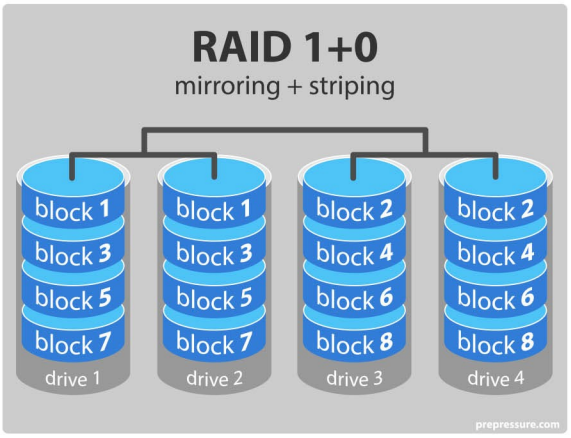
به این معنا که اگر یک دیسک دچار مشکل شود به کمک (حداقل) دو دیسک دیگر می توان اطلاعات از دست رفته را بازگردانی کرد. همانطورکه میدانید انواع مختلفی از مکمل سازی (زوج و فرد) وجود دارد و با توجه به همین موضوع انواع مختلفی از این نوع رِید وجود دارد؛ برای مثال RAID6 از دو دیسک مکمل استفاده میکند.



توجه داشته باشید که در عمل هیچ گاه یک دیسک را به عنوان دیسک مکمل در نظر نمیگیرند و اطلاعات مکمل را در تمامی دیسک ها پخش میکنند. چرا که اگر در روش اول

دیسک مکمل دچار اشکال شود راهی برای بازگردانی اطلاعات وجود ندارد. سرعت خواندن در این نوع از رِید بالا می باشد ولی سرعت نوشتن بستگی به دیسک ها دارد.

انواع دیگری از رِید نیز وجود دارند که از ترکیب انواع ذکر شده به وجود می آیند برای مثال RAID1+0 که اطلاعات را به دو بخش تقسیم کرده و در دو دیسک ذخیره میکند سپس همین اطلاعات را دوباره روی دو دیسک دیگر ذخیره میکند. یعنی روی هم رفته چهار دیسک داریم. (ابتدا RAID0 انجام میشود سپس RAID1).



به طور خلاصه می توان گفت رِید روشی است برای ذخیره اطلاعات به گونه ای که بازیابی آن آسان تر باشد.

در آخر جدولی که هر سه نوع رِید به همراه خلاصه ای از وضعیت آنها آورده شده است.

RAID	0 Strip	1 Mirror	5 Parity
Fault-Tolerance	No	Yes	1 Disk Max
Reading Speed	High	Disk Depen dent-	High
Writing Speed	High	Low	Disk Depen -dent
Min Dsik	2	2	3
Complemen -tary Disk	No	No	Yes

3

مقدمات JavaFx

جاوا مثل زبان های برنامه نویسی دیگر که در هسته اصلی خود تنها بر محور خط فرمان کار میکنند می باشد اما با اضافه کردن کتابخانه های مشخصی (مثل: javafx) می توان برنامه ای با رابط کاربری گرافیکی طراحی نمود.

امیرعلی گرجی

JavaFX چیست و چه کار بردهایی دارد ؟-

JavaFX کتابخانه ای برای توسعه گرافیکی برنامه های نوشته شده به زبان Java می باشد که ۱۲ سال بعد از Swing منتشر شد.

این پلتفورم شبیه به adobe flex (یک SDK برای نوشتن internet application و برنامه های cross-platform) و Microsoft Silverlight (مشابه adobe flex با این تفاوت که Silverlight یک framework می باشد) بوده و تقریبا در یک گروه بندی قرار میگیرند با این تفاوت که adobe flex و Microsoft Silverlight برای توسعه گرافیکی از xml استفاده می کنند ، درابتدا javaFX از طرف شرکت sun با همان شکل و شمایل swing یعنی اسکریپت نویسی های رایج شروع کرد که اگر با component های گرافیکی swing آشنا باشید مهاجرت به javaFX برای شما کار سختی نخواهد بود ، اما از سال ۲۰۱۱ زبان اختصاصی javaFX برای قسمت گرافیکی برنامه با نام fxml منتشر شد که فایل های مربوط به این زبان با پسوند fxml. نمایش داده می شوند. با این کار این امکان وجود دارد که در دو محیط کاملا جدا اقدام به توسعه گرافیک برنامه (بوسیله fxml) و منطق برنامه (با زبان java) بکنیم.

از Script استفاده کنم یا fxml؟

در ابتدای یادگیری این تکنولوژی برای توسعه دهنده این مسئله پیش می آید که آیا یادگیری fxml لازم است یا فراگیری script ها کافی است.

دو دلیل وجود دارد که از fxml استفاده کنیم :

۱) استفاده از fxml این توانایی را به شخص می دهد که کد های نوشته شده به زبان جاوا را از بخش گرافیکی جدا کند و در هر بخش جداگانه دست به توسعه برده شود.

۲) زمانی که توسعه یک پروژه کامل شده و نسخه اولیه منتشر می شود بعد از مدتی ممکن است برنامه شما نیاز به بروز رسانی هایی در بخش رابط کاربری داشته باشد ، با استفاده از fxml شما حتی لازم نیست به یک خط کد جاوا برای تغییر جلوه برنامه دست بزنید و تنها با تغییر component ها این امکان برای توسعه دهنده فراهم می شود که منطق اصلی برنامه خود را با رابط کاربری نوینی عرضه کند.

در هر دو روش امکان استفاده از فایل های css برای برنامه نوشته شده با پلتفرم javaFX وجود دارد و امکانات خیلی مفیدی را در اختیار توسعه دهنده قرار می دهد.

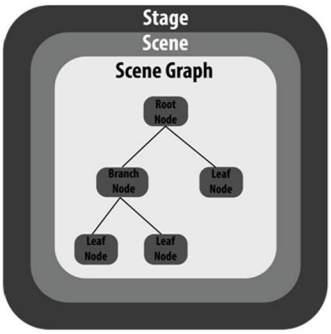
معماری javaFX را Oracle چگونه طراحی کرده است ؟

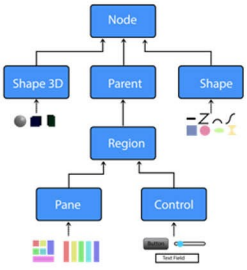
در مقدماتی ترین قسمت از یک برنامه javaFX کلاس abstract به نام Application وجود دارد که تابع main از آن ارث می برد، این کلاس تابعی به نام start درخود دارد که بدنه آن توسط توسعه

توسعه دهنده پیاده سازی می شود. در راس هر پنجره گرافیکی که کاربر بعد از اجرا کردن برنامه می بیند کلاسی به نام Stage هست که نمونه از این کلاس به تابع start به عنوان پارامتر ورودی داده میشود.

با استفاده از fxml شما حتی لازم نیست به یک خط کد جاوا برای تغییر جلوه برنامه دست بزنید

همچنین هر Stage محتویات خود را در یک Scene به نمایش میگذارد. برای درک بهتر این سلسله مراتب یک سالن تئاتر را فرض کنید که خود سالن Stage و هر صحنه ای که بازیگر ها در آن بازی میکنند Scene هست که مرتبا امکان عوض شدنش در یک نمایش تئاتر وجود دارد مثلا تئاتری را فرض کنید که با صحنه ای از داخل منزل شروع می شود و این تغیر صحنه ها تا پایان نمایش ادامه دارد؛ همان گونه که در داخل یک صحنه بازیگرها، صندلی و سایر وسایل قرار می گیرند در معماری javaFX گرافی با عنوان scene graph وجود دارد که تمامی المان های گرافیکی را در خود جای می دهد.





از کجا شروع کنم ؟

تا jdk۸، شرکت oracle کتابخانه های مربوط به javaFX را همراه jdk در اختیار کاربران قرار می داد. اما همزمان با انتشار jdk های بعدی از سوی این شرکت این دو از یکدیگر جدا شدند. برای همین در قدم اول بعد از نصب jdk (که برای منطق برنامه از آن استفاده می شود) ، دانلود و اضافه کردن کتابخانه javaFX است. برای دانلود به این لینک بروید. بعد از دانلود کتابخانه javaFX برای اضافه کردن api های آن به intelij از این راهنما استفاده کنید. در مرحله بعد باید در فایل ماژول مربوط به پکیج اعلام کنید که این ماژول به javaFX نیازمند است، برای اینکار روی src راست کلید کرده و از بخش new گزینه module-info را انتخاب کرده و محتویات آن را به صورت زیر تغییر دهید.

Main.java | module-info.java | Cont

```
module TempFX2 {

    requires javafx.controls;
    requires javafx.fxml;
    opens sample;
}
```

در این نمونه نام پروژه TempFX2 و نام پکیجی که فایل های java. و fxml. در آن قرار دارد sample می باشد. یک پروژه ساده برای آشنایی بیشتر شما دوستان از طریق این لینک در github در دسترس می باشد.

با هر بار نمایش tooltip دکمه مورد نظر ، جمله"this is a ToolTip" در کنسول چاپ می شود. و با کلیک بر روی دکمه ، جمله "ZeroAndOne Journal" نمایش داده خواهد شد. به منظور نمایش تصویر پس زمینه از css داخل style = "" استفاده شده است.

سخن پایانی

منابع بسیار زیادی برای یادگیری این کتابخانه در سطح اینترنت وجود دارد و به شخصه پیشنهاد می کنم اگر برایتان مقدور هست از ویدیو های youtube استفاده کنید.

وبسایت هایی مثل udemy آموزش هایی حول این کتابخانه دارند که از کیفیت بالایی برخوردار هستند.

با یادگیری این کتابخانه دیگر محدود به فضای console در برنامه های جاوا نخواهید بود و میتوانید قابلیت های خودتان را در زمینه این زبان برنامه نویسی که امروزه خواهان بسیاری دارد بالا ببرید.

دلیل اهمیت javaFX این است که از سوی شرکت oracle در حال توسعه بوده و در گذر زمان قابلیت های آن افزایش می یابد که این مهم در swing وجود ندارد.

نویسنده لوپ این شماره :
پیام رنجبر

معرفی کتاب

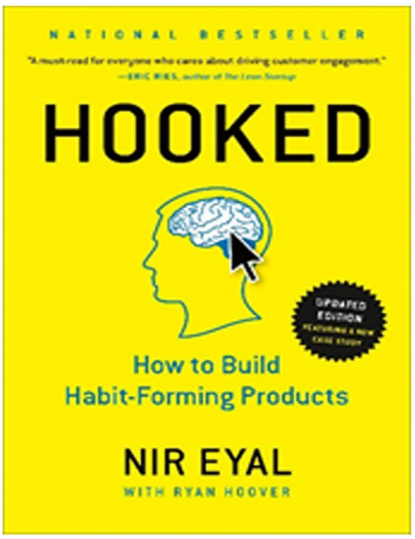
در این قسمت از بخش سریالی معرفی کتاب، به کتاب مهمی که میتواند مکمل تاثیر گذاری در ارتقای مهارت هایتان در کنار رشته تحصیلی شما باشد، نگاه کوتاهی می اندازیم. شما نیز از این پس میتوانید کتاب هایی که مناسب میدانید را در این نشریه با دیگران به اشتراک بگذارید و به ترویج فرهنگ کتاب خوانی کمک کنید. برای اینکار میتوانید از طریق راه های ارتباطی انجمن علمی مهندسی کامپیوتر با نشریه در ارتباط باشید.

مرا وادار به فکر کردن نکن

Don't make me think |

همانطور که از نام این کتاب مشخص است، کتاب به بررسی مسائل مربوط به ارتباط انسان و نرم افزار ها و یا به عبارت بهتر تجربه کاربری (User Experience) آن ها می پردازد.

نویسنده کتاب آقای Steve Krug با زبانی ساده مفاهیم مربوط به ارتباط ذهن مخاطب با محصول نرم افزاری شما را بیان میکند و به جنبه های مهم و تاثیر گذار یک رابط کاربری مناسب برای وب¬ سایت های مختلف همراه با نمونه های متعدد می پردازد. بیشتر مفاهیم اشاره شده در کتاب حول وب سایت ها و وب اپلیکیشن ها می چرخد اما اطلاعات مفیدی که در اختیار خوانند قرار میگیرد پایه ای هستند و قابل استفاده در هر نرم افزار و محصولی میباشند. اگر میخواهید مفاهیم پایه ای طراحی یک تجربه کاربری خوب را در مدت کوتاهی فرا بگیرید، این کتاب بهترین انتخاب برای شماست.



به دام افتاده

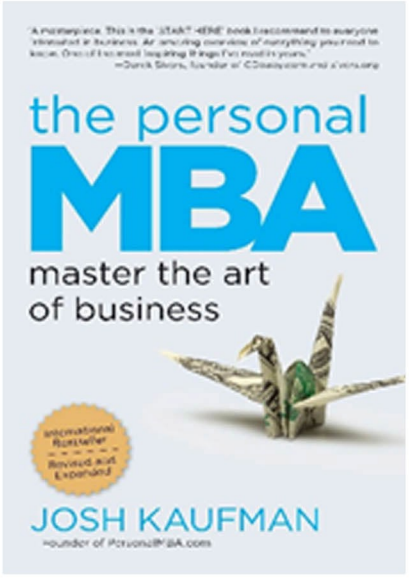
Hooked: How to Build | Habit-Forming Products

چگونگی استفاده مفاهیم روانشناسی در طراحی محصولات نرم افزاری موضوعی است که با جزئیات و نمونه های فراوان در این کتاب پرداخته میشوند. با بررسی طراحی روانشناختی و تجربه کاربری محصولات اعتیاد آور مانند شبکه های اجتماعی نویسنده الگو ها و مفاهیم تاثیر گذار در آن را برای خوانندگان بازگو میکند. بررسی سیستم امتیاز دهی مغز انسان نسبت به عوامل بیرونی و درونی ذهن برای طراحی یک محصول موفق و همه گیر از جمله موضوعات این کتاب می باشد. از ویژگی های مهم شیوه بیان ساده و گویا Nir Eyal، نویسنده کتاب است، که مسائل پیچیده ای مانند طراحی محصول بر اساس خصوصیات رفتاری مخاطبان را واکاوی میکند. این کتاب توسط انتشارات آریانا قلم تحت عنوان "قلاب: چگونه محصولی بسازیم که مخاطب را شبانه روزی درگیر کند" ترجمه و انتشار یافته است.

در جملاتی از این کتاب می خوانیم :

"

ماتریس اعمال نفوذ ابزار تصمیم‌گیری ساده‌ای است که برای کارآفرینان، کارمندان و سرمایه گذاران طراحی شده است. هدف ماتریس اعمال نفوذ مشخص کردن کسب و کار اخلاقی موفق و همچنین چگونگی شکل دادن به عادت نیست.



MBA شخصی شما | The Personal MBA: Master the Art of Business

MBA یا مدیریت ارشد کسب کار یکی از رشته های پرطرفدار کارشناسی ارشد است. هر ساله افراد زیادی با پیش زمینه های مهندسی و فناوری وارد آن میشوند. با این حال ممکن است افراد زیادی همانند نویسنده این کتاب قادر به گذراندن این دوره آموزشی نباشند. Josh Kaufman نویسنده، براساس سال ها تجربیات خود، در این کتاب به بررسی مفاهیم پایه ای مدیریت کسب و کار با بیانی شیوا و به صورت فرمت بندی شده می پردازد. اگر دنبال بالا بردن درک بیزینسی و اقتصادی خود و شروع یک کسب و کار در رشته خودتان هستید این کتاب منزل گاه عالی برای شروع است. مسائلی مانند کار آفرینی، مدیریت، رهبری، اقتصاد و... از جمله مواردی هستند که در فصل های مختلف این کتاب مطرح و تشریح شده اند.

“ Life isn't about finding yourself,
Life is about creating yourself. ”

— George Bernard Shaw