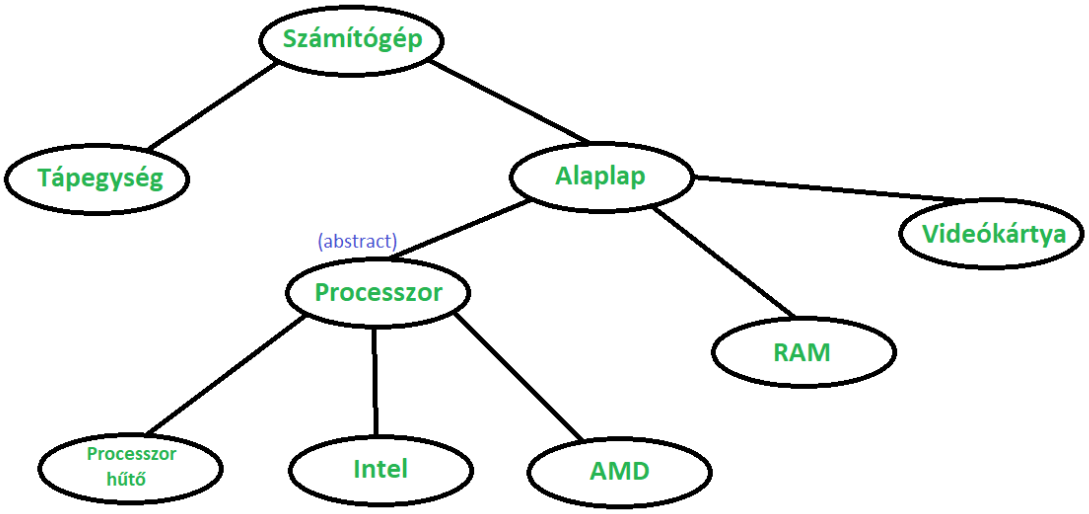
Konfiguráció összeállító rendszer

A program egy számítógép konfigurációt állít össze. Indításkor egy menü fogadja a felhasználót, navigálással és a képernyőre kiírt utasítások segítségével össze lehet állítani egy egyedi konfigurációt. Lehetőség van arra is, hogy automatikusan összeszedje a rendszer a legolcsóbb illetve az átlagosan legjobb minőségű konfigurációt. Minden esetben választani kell egy alaplapot, hisz ettől az elemtől sok más komponens kompatibilitása függ.

Osztályhierarchia:



Minden hardware megvalósítja az IHardverElem interface-t. Ezeknek a konstruktorában be vannak állítva a megfelelő értékek.

class Számítógép:

Ez az osztály vizsgálja, hogy minden komponens be van-e szerelve, kompatibilisek-e egymással, illetve összefoglalja a konfiguráció összértékét és a komponensek átlagos minőségét. Egy konfiguráció építésénél a tápegységet mindig utoljára építjük be, ezt is ellenőrzi.

class PSU:

Minden gép üzemeltetéséhez szükség van tápegységre. Minden egyes hardwarenek van valamiféle fogyasztása (kivéve az Alaplap példányoknak, ezekben az esetekben a fogyasztás elhanyagolható). Mivel a tápegységet legutoljára építjük be, így ismerjük a Számítógép *SumPowerUsage* adattagja által tárolt összfogyasztást. Amennyiben a tápegység kapacitása nem elegendő, a komponens „elromlik”, amit egy *InsufficientPower* esemény jelez a felhasználó számára, melyben megtudjuk a hiány mértékét is. Amennyiben két tápegységet próbálunk beszerelni, a program DuplicateItemException típusú kivételt dob.

class Alaplap:

Az alaplap az összes további komponens alapja, ez előtt nem szerelhető be semmi sem. A kompatibilitást ellenőrizve rendelkezik *Socket* és *Datarate* tulajdonságokkal, melyek enum felsorolás által rögzített értékek lehetnek. A felsorolások első tagja egy *Null* érték, hiszen egy leszármaztatott

Processzor osztály például nem rendelkezik olyan tulajdonsággal, hogy milyen típusú memória. Ennek megfelelően a konstruktor alapértéke ez a *Null* lesz. A konstruktorban meg lehet adni azt is, hogy hány RAM modult illetve hány videókártyát lehet behelyezni. Ez az alkatrész minden esetben kompatibilis a géppel, és sosem romlik el. A Beépít(pc) metódus virtuális, minden alosztály ezt a metódust fogja meghívni. A program DuplicateItemException típusú kivételt dob, amennyiben két alaplapot próbálunk beszerelni.

class Processzor:

Absztrakt osztály, beépítésnél a program megvizsgálja, hogy kompatibilis-e az alaplappal. Amennyiben nem, a beépítés semmissé válik, és elromlik az alkatrész, amit egy *FalseSocket* esemény jelez a felhasználó számára. Ha két processzort próbálunk beszerelni, a program ezt DuplicateItemException típusú kivételt dob. Szintén kivételt kap a felhasználó, ha a processzort az alaplap előtt próbálja beszerelni (WrongOrderException). Az IntelProcesszor és AMDProcesszor osztályok csak példányosítás miatt vannak, egyébként az ős konstruktorát hívják meg.

class CPUHuto:

Processzor hűtő osztály, csakis azután lehet beszerelni, ha alaplap és processzor is van már. Amennyiben két hűtőt szerelnénk be, a program DuplicateItemException típusú kivételt dob. Beépítésnél amennyiben nem kompatibilis a hűtő a processzorral, a beépítés semmissé válik és a hűtő elromlik, melyről a felhasználó *NotCompatibleCooler* esemény által értesül.

class RAM:

Csak az alaplap után lehet beszerelni, különben kivételt dob a program. Egyszerre több RAM modult is be lehet építeni, viszont ha az alaplap már nem képes több befogadására, a program DuplicateItemException típusú kivételt dob. Amennyiben nem kompatibilis az alaplappal, a beépítés semmissé válik és a felhasználó erről egy *FalseDDR* esemény által értesül.

class GPU:

Az alaplap után lehet csak beszerelni, a RAM osztályhoz hasonlóan egyszerre több videókártyát is beépíthetünk, viszont ha már nem fér el az alkatrész, akkor DuplicateItemException típusú kivételt kap a felhasználó. A videókártya nem romlik el.

class ConfigCreator:

Ebben az osztályban található a menü és annak működése, mely az eddig bemutatott osztályokkal dolgozik. Itt le van képezve egy gráf, valamint található egy lista is, amely a beépített komponenseket tárolja el. A *Komponensek* fül alatt találhatóak a leképzett IHardverElem interface-t megvalósító példányok.

- Start() – Az egyetlen publikus metódus, melynek a Main-ből történő meghívása során leképez egy PC-t, és meghívja a Menu(PC) metódust.
- Menu(PC) – A *UI* fül alatt található maga a menü struktúrája, az egyes komponenseket billentyű lenyomásával tudjuk beszerelni. Az eddig beszerelt elemeket a „*Kosár tartalma*” alatt látja a felhasználó, melyek egy listában vannak eltárolva. Ezen kívül lehet látni a kosár összértékét, valamint az átlagos minőséget is. Amennyiben a működéshez már elegendő alkatrész van beszerelve, a program erről értesíti a felhasználót.
  Maga a menü egy *do-while* ciklussal működik, a megfelelő billentyűk lenyomásával az adott menüpontot hívjuk meg. Amennyiben nem optimális keresést választ a felhasználó, akkor

előfordulhat, hogy egymással nem kompatibilis komponenseket próbál beszerelni. Az ilyen esetekre fel van készítve a program, hogy elkapja ezeket a kivételeket. *ESC* billentyű lenyomásával a program futása véget ér.

- MB(PC) – Ezek közül az alaplapok közül választhat a felhasználó. Amennyiben a felhasználó hozzáad egy alaplapot a konfigurációhoz, a Beépít(PC) metódus ezt beépíti, és megvizsgálja, hogy az adott feltételeknek eleget tesz-e az adott alkatrész (pl. kompatibilis-e, van-e már beszerelt alaplap, stb.). Ha a beépítés sikeres volt, a beépített elemeket tartalmazó listához hozzáadja a kiválasztott komponenst. Végül egy ComponentAdded(item) metódus jelzi a felhasználó számára, hogy mit is rakott bele a kosárba. Amennyiben a felhasználó meggondolja magát, *Backspace*-el vissza tud lépni a főmenübe. Bármely más gomb lenyomására a program újra megjeleníti ezt az almenüt. A továbbiakban felsorolt metódusok hasonló módon működnek, egy-két apróbb változtatással.
- CPU(PC) – Ezek közül a processzorok közül választhat a felhasználó. Tekintve, hogy egy processzornál már inkompatibilis eset is felmerülhet (például egy *Intel* típusú alaplapba egy *AMD* processzort kíván behelyezni a felhasználó), minden példány *FalseSocket* eseményére fel van iratkozva a CPUCompatibilityIssue metódus, mely meghívása esetén tájékoztatja a felhasználót a hibájáról.
- CPUCooler(PC) – Ezek közül a processzor hűtők közül választhat a felhasználó. Az esetleges inkompatibilis eseteket a CPUCoolerCompatibilityIssue metódus kezeli, mely fel van iratkozva minden példány *NotCompatibleCooler* eseményére.
- RAM(PC) – Ezek közül a RAM modulok közül lehet választani. Amennyiben nem illeszkedik az alaplapba, a program a RAMCompatibilityIssue metódussal jelzi ezt a felhasználó számára (ami természetesen fel van iratkozva minden példány *FalseDDR* eseményére).
- GPU(PC) – Ezek közül a videókártyák közül lehet választani.
- PSU(PC) – Ezek közül a tápegységek közül lehet választani. Ezt a program csakis akkor engedi beszerelni, ha minden más komponensből már legalább egy példány be van szerelve. A menü jelzi, hogy az eddig beszerelt komponenseknek mennyi az összfogyasztása, ennek megfelelően lehet tápegységet választani. Amennyiben mégse lenne elegendő, a program ezt egy InsufficientPowerIssue metódussal jelzi, mely fel van iratkozva minden példány *InsufficientPower* eseményére.
- BargainSearch() – Ez a metódus a legolcsóbb konfigurációt állítja össze. Csak akkor lehet meghívni, ha még nincs működő konfiguráció összeállítva. Meghíváskor a program felépít egy gráfot, melynek csúcsai az egyes komponensek, és az egymással kompatibilis eszközök között fut él. Futtatáskor a felhasználónak csak az alaplapot kell kiválasztania, hisz esetleges későbbi bővítésre van lehetőség (például a felhasználó még szeretne RAM-ot elhelyezni). Ennek megfelelően alaplapokat nem, de minden más komponenst vizsgál a program. A vizsgált példányok árait elhelyezi egy-egy listába, melyeket továbbad a gráf Melysegibejaras() metódusnak. A továbbadott adattagok közül a which egész szám típusú adattag azt jelzi a metódusnak, hogy a mélységi bejárás során a megfelelő szempontokat figyelembe véve válassza ki a komponenseket (ebben az esetben a legolcsóbb elemeket szűrje ki).
- QualitySearch() – Ez a metódus az átlagosan legjobb minőségű konfigurációt állítja össze. A BargainSearch() metódushoz hasonlóan csak nem működő konfiguráció esetén lehet meghívni, és ugyanúgy ki kell választani az alaplapot. A vizsgált példányok minőségét elhelyezi egy-egy listába, melyeket továbbad a gráf Melysegibejaras() metódusnak. A továbbadott adattagok közül a which egész szám típusú adattag azt jelzi a metódusnak, hogy a mélységi bejárás során a megfelelő szempontokat figyelembe véve válassza ki a komponenseket (ebben az esetben a legjobb minőségű elemeket szűrje ki).

class Graf<T>:

Gráf felépítésére szolgáló osztály. Az optimális keresést megvalósító metódusokat is tartalmazza.

- Melysegibejaras() – MelysegibejarasRek() metódusnak állítja elő a listát, és meg is hívja.
- MelysegibejarasRek() – Rekurzívan mélységi bejárást végez a gráfon, vagy ha létezik which adattag, akkor az azon keresztül definiált metódust hívja meg.
- BargainSetup() – Az egyes típusokon belül kiszűri azt az elemet, amely kompatibilis a kiválasztott alaplappal és az így kapott kategórián belül a legolcsóbb. A tápegységek közül nem vizsgál árat, itt azt a példányt választja ki, melynek kapacitása elegendő a konfiguráció üzemeltetéséhez.
- QualitySetup() – Az egyes típusokon belül kiszűri azt az elemet, amely kompatibilis a kiválasztott alaplappal és az így kapott kategórián belül a legjobb minőségű. A tápegységek közül nem vizsgál minőséget, itt azt a példányt választja ki, melynek kapacitása elegendő a konfiguráció üzemeltetéséhez.