

Jälkipuhe

Jos olet edelleen nälkäinen C++-kielen ohjeistoille nautiskeltuasi 50 tapaa kehittää ohjelmiasi ja työtapojasi, olet ehkä kiinnostunut toisesta aiheeseen liittyvästä kirjastani, *More Effective C++: 35 New Ways to Improve Your Programs and Design*. Kuten *Effective C++*, *More Effective C++* kattaa materiaalin, joka on olennainen tehokkaaseen C++-kielen ohjelmistokehitykseen, mutta *Effective C++* keskittyy enemmän perusasioihin, siinä missä *More Effective C++* kuluttaa enemmän aikaa kielen uusimpiin piirteisiin ja ohjelmoinnin edistyneempiin tekniikkoihin.

Voit löytää yksityiskohtaista tietoa kirjasta *More Effective C++* — sisältäen neljä kokonaista Kohtaa, kirjan luettelon suositeltavasta luettavasta, ja lisää — kirjan *More Effective C++* web-sivulla osoitteessa: <http://www.awl.com/cp/mec++.html>. Siinä tapauksessa, että et jaksaa odottaa, kirjan *More Effective C++* sisällöstä on yhteenveto alla.

Perusteet

- Kohta 1: Tee ero osoittimien ja viittausten välillä
- Kohta 2: Suosi C++-tyylisiä muunnoksia
- Kohta 3: Älä koskaan kohtelee taulukoita polymorfisesti
- Kohta 4: Vältä vastikkeettomia oletusmuodostinfunktioita

Operaattorit

- Kohta 5: Ole tietoinen käyttäjän määrittelemistä muunnosfunktioista
- Kohta 6: Tee ero lisäys- ja vähennysoperaattoreiden etu- ja takaliitemuotojen välillä
- Kohta 7: Älä koskaan kuormita `&&`, `|`, tai `,`
- Kohta 8: Ymmärrä operaattoreiden `new` ja `delete` eri tarkoitukset

Poikkeukset

- Kohta 9: Käytä tuhoajafunktioita estääksesi resurssien vuodot
- Kohta 10: Estä muodostinfunktioiden resurssivuodot

- Kohta 11: Estä poikkeukset, jotka aiheutuvat tuhoajafunktioista poistuttaessa
- Kohta 12: Ymmärrä se, kuinka poikkeuksen muodostaminen eroaa parametrin välittämisestä tai virtuaalifunktion kutsumisesta
- Kohta 13: Ota poikkeukset kiinni viittauksella
- Kohta 14: Käytä poikkeusten määrittelyitä ymmärtäväisesti
- Kohta 15: Ymmärrä poikkeuskäsittelyn kustannukset

Tehokkuus

- Kohta 16: Muista 80-20 -sääntö
- Kohta 17: Harkitse laiskan evaluoinnin käyttämistä
- Kohta 18: Lyhennä odotetuista laskutoimituksista koituvien kustannusten velka
- Kohta 19: Ymmärrä tilapäisten olioiden alkuperä
- Kohta 20: Helpota paluuarvon optimointia
- Kohta 21: Kuormita välttääksesi implisiittisest tyyppimuunnokset
- Kohta 22: Harkitse $op=$ -funktion käyttöä itsenäisen op -funktion sijasta
- Kohta 23: Harkitse vaihtoehtoisia kirjastoja
- Kohta 24: Ymmärrä virtuaalifunktioiden, moniperinnän, virtuaalisten kantaluokkien ja RTTI:n kustannukset

Tekniikat

- Kohta 25: Muodostinfunktioiden ja ei-jäsenfunktioiden virtualisointi
- Kohta 26: Luokassa olevien olioiden määrän rajoittaminen
- Kohta 27: Kekopohjaisten olioiden edellyttäminen tai estäminen
- Kohta 28: Viisaat osoittimet
- Kohta 29: Viittausten laskeminen
- Kohta 30: Proxy-luokat
- Kohta 31: Funktioiden määrittäminen virtuaalisiksi useampaan kuin yhteen olioon nähden

Sekalaista

- Kohta 32: Ohjelmoi futuurissa
- Kohta 33: Tee epälehtiluokista abstrakteja
- Kohta 34: Ymmärrä kuinka yhdistät C++- ja C-kielen samassa ohjelmassa
- Kohta 35: Perehdy kielen standardiin

Suosittelavaa luettavaa

`auto_ptr`-toteutus