Generics

to save the day

```kotlin
val account = SavingsAccount("GB00...", 10_000.toBigDecimal(), "0.024".toBigDecimal())
println(account.interest)

val (updatedAccount, _) = account.withdraw(10.toBigDecimal())
println(updatedAccount.interest)
```

```
val account = SavingsAccount
        account.interest

val  updatedAccount       = account.withdraw
        updatedAccount.interest
```

f

u

n

<

T

A

C

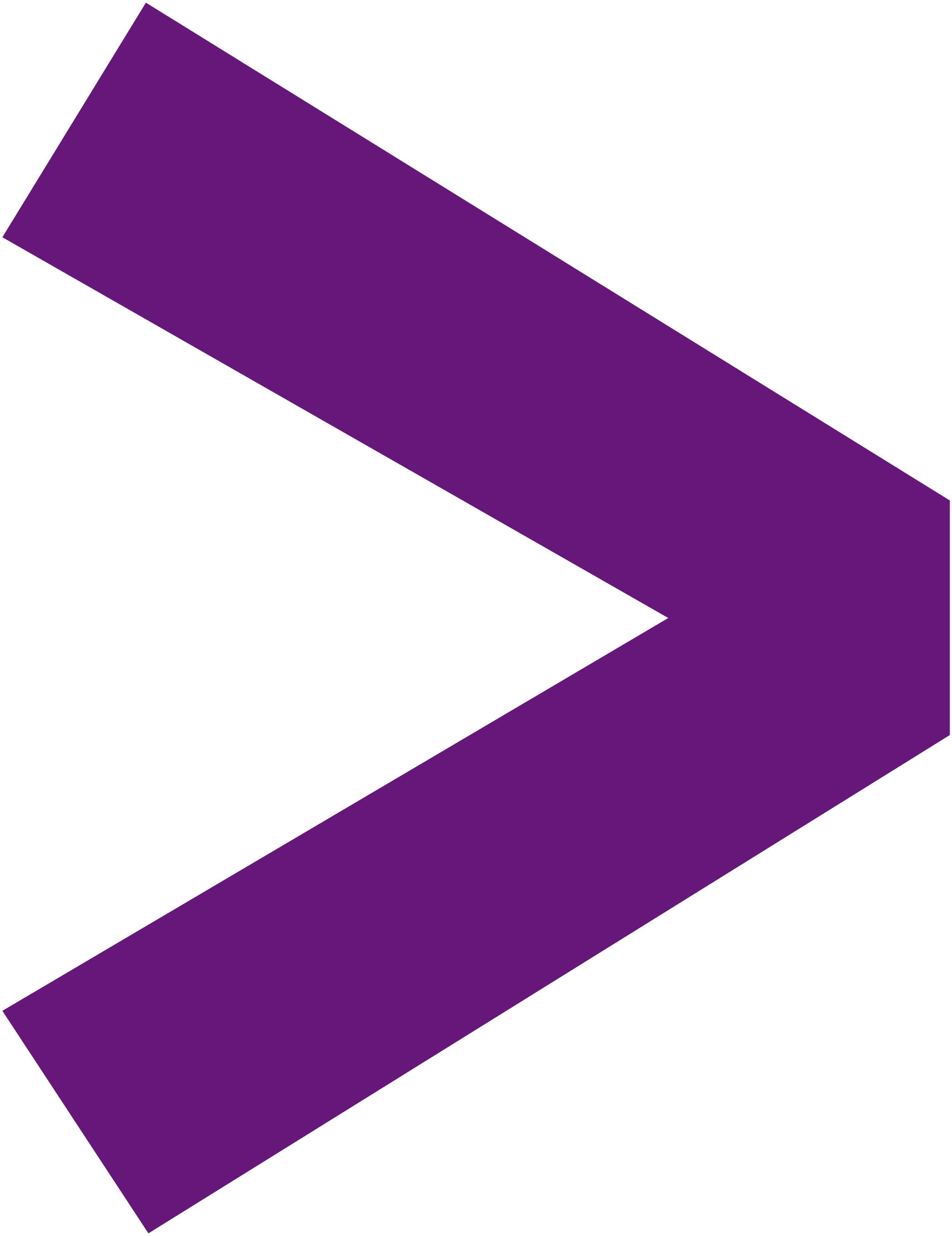
C

O

u

n

t



T

●

W

i

t

h

d

r

a

W

a

m

O

u

n

t

●

●

B

i

g

D

e

C

m

a

Լ

)

●

●

P

i

r

,

B

i

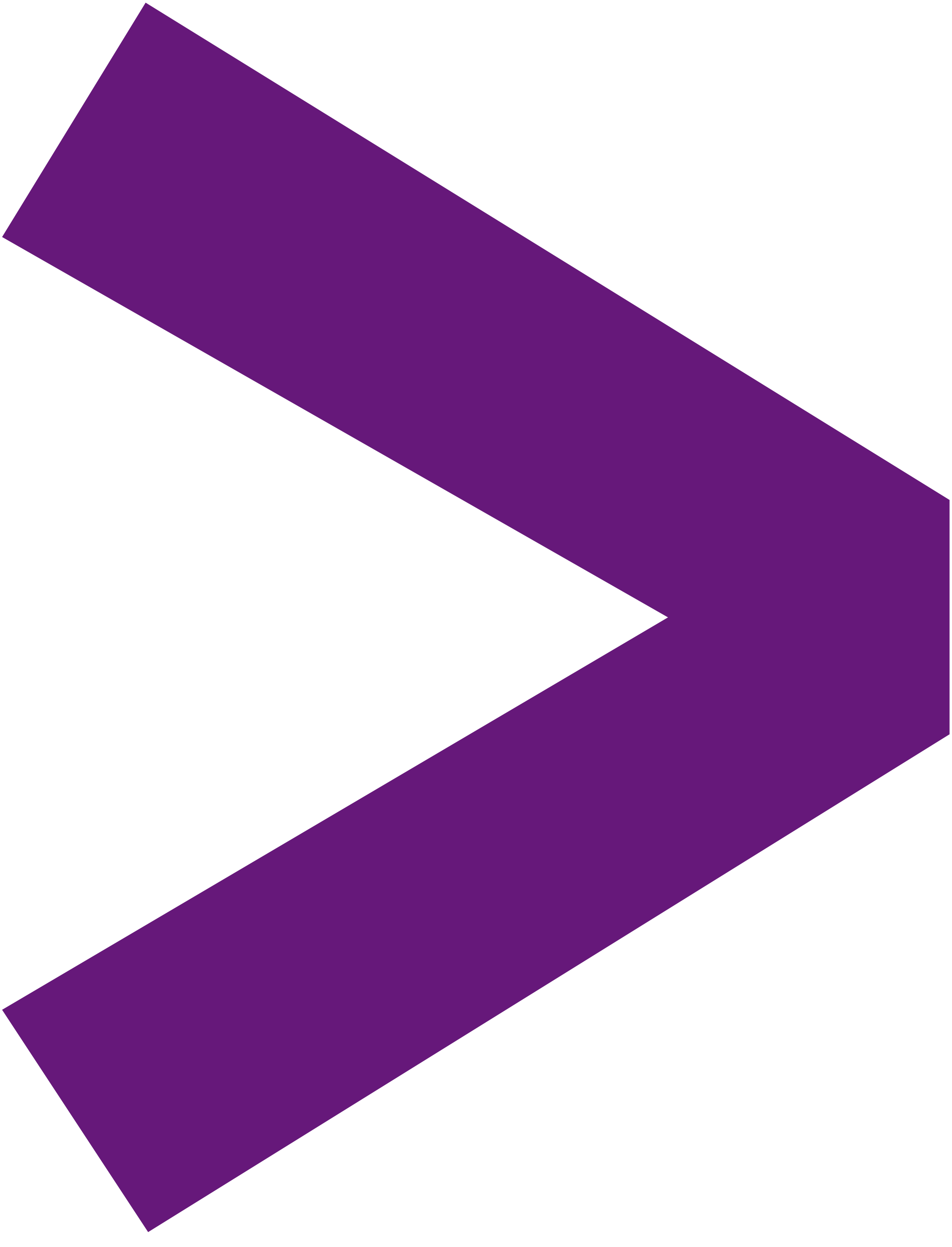
g

D

e

C

ʅ



[redacted]

[redacted]

r

{

V

a

ղ

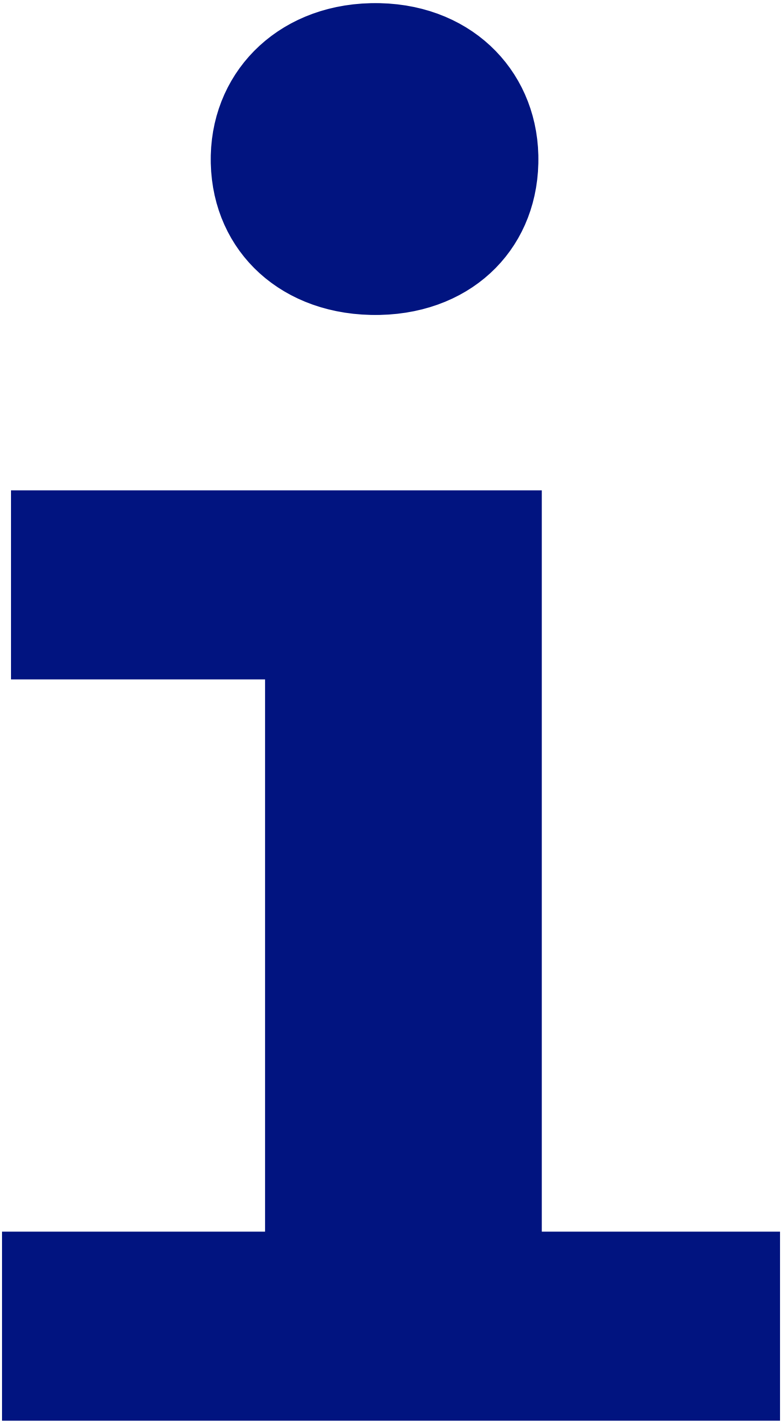
a

C

[redacted]

t

h



S

a

S

A

O

U

n

W

h

e

(

O

u

n

t

{

i

C

r

e

A

[redacted]

>

a

C

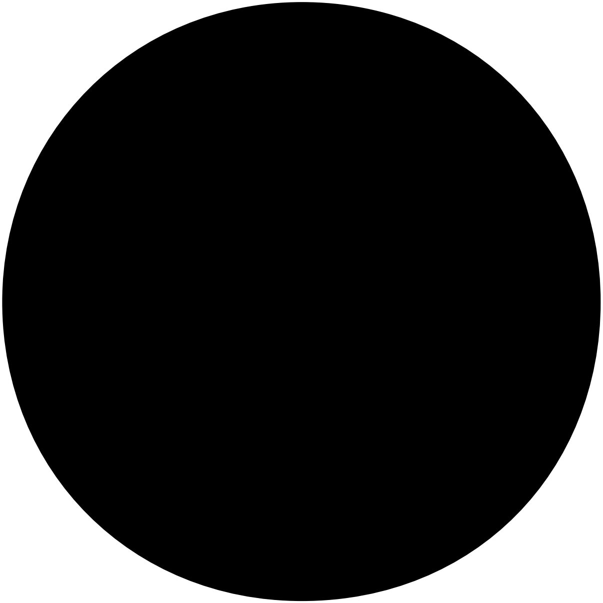
C

O

u

n

t



p

y

(

b

a

l

a

n

C

e

$=$

b

a

Ն

a

n

C

e

[redacted]

a

m

u

O

u

n

t

S

V

g

S

p

y

*

O

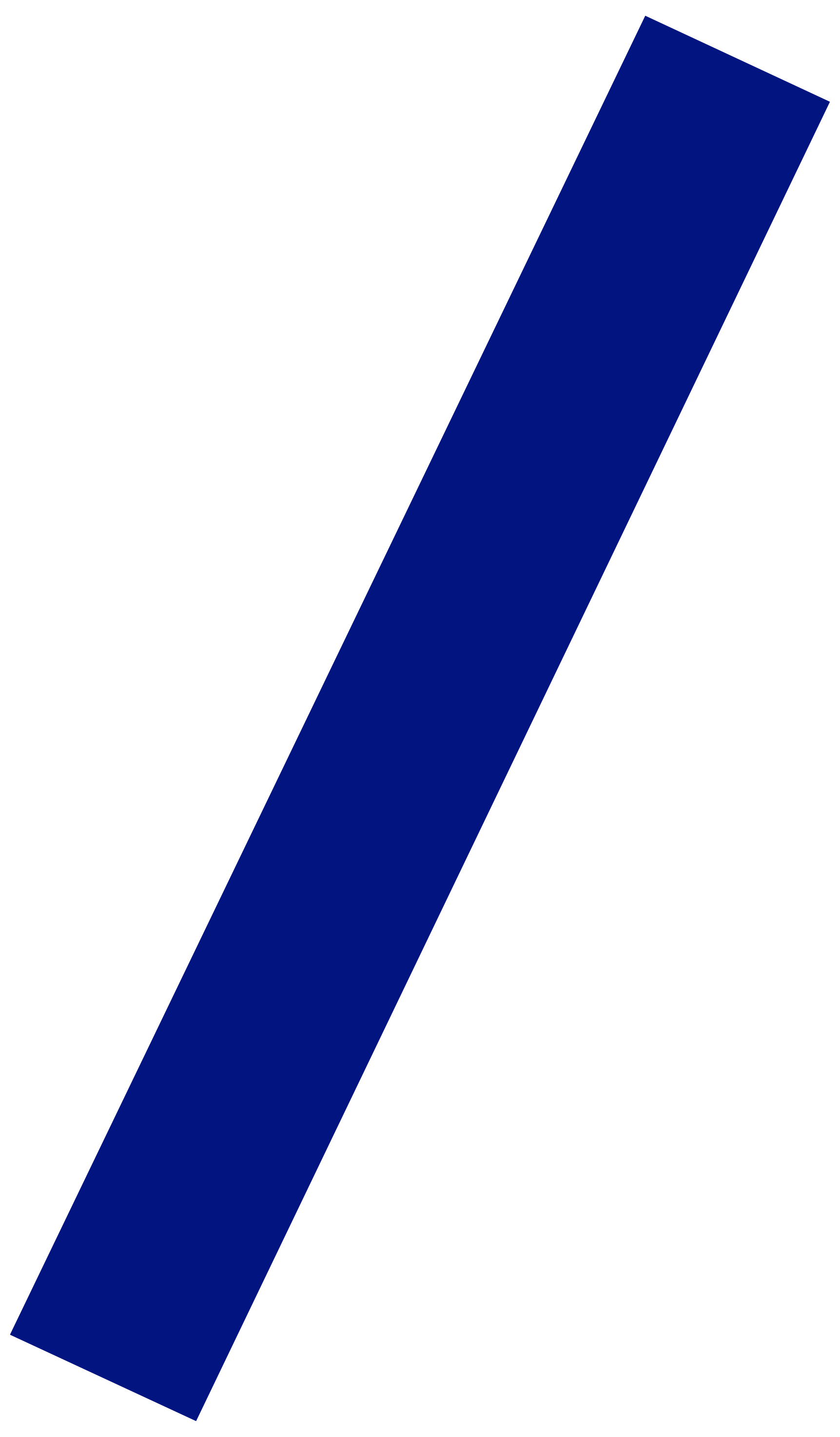
m

m

i

t

d



}

}

```
fun <T : Account> T.withdraw(amount: BigDecimal): Pair<T, BigDecimal>
```

```kotlin
fun <T : Account> T.withdraw(amount: BigDecimal): Pair<T, BigDecimal> = run {
    val account = this as Account
    when (account) {
        is CurrentAccount -> account.copy(balance = balance - amount) as T to amount
        is SavingsAccount -> account.copy(balance = balance - amount) as T to amount
        /* ommited */
    }
}
```

Generics

to save the day

```kotlin
fun <T : Account> T.withdraw(amount: BigDecimal): Pair<T, BigDecimal> = run {
    val account = this as Account
    when (account) {
        is CurrentAccount -> account.copy(balance = balance - amount) as T to amount
        is SavingsAccount -> account.copy(balance = balance - amount) as T to amount
        /* ommited */
    }
}

val account = SavingsAccount("GB00...", 10_000.toBigDecimal(), "0.024".toBigDecimal())
println(account.interest)

val (updatedAccount, _) = account.withdraw(10.toBigDecimal())
println(updatedAccount.interest) ✓
```

Generics

type erasure