```
randuniform(dims::Base.Dims) = randuniform(Float64, dims)
randuniform(::Type{T}, dims::Base.Dims) where {T<:Number} = rand(T, dims)

randnormal(dims::Base.Dims) = randnormal(Float64, dims)
randnormal(::Type{T}, dims::Base.Dims) where {T<:Number} = randn(T, dims)

randisometry(dims::Base.Dims{2}) = randisometry(Float64, dims)
randisometry(::Type{T}, dims::Base.Dims{2}) where {T<:Number} =
    dims[1] >= dims[2] ? _leftorth!(randnormal(T, dims), QRpos(), 0)[1] :
        throw(DimensionMismatch("cannot create isometric matrix with dimensions
$dims; isometry needs to be tall or square"))

const randhaar = randisometry
```