# Herná logika

## Spôsob generovania herného poľa

V hernej slučke sa v každnom kole vygeneruje 8 nových inštancií triedy **Hole**. 4 z nich sa ukladajú do poľa **playerHoles** a ďalšie 4 do poľa **evaluationHoles**. Kolo sa končí, keď hráč priradí každému zo 4 objektov triedy **Hole** v poli **playerHoles** jednu z hodnôt enumerácie **Pin**, potvrdí svoj výber a počítač vyhodnotí jeho tip priradení jednej z hodnôt enumerácie **Pin** do každého zo 4 objektov typu **Hole** v poli **evaluationHoles**.

### Spôsob overovania stavov hry

Po každom kole sa overí, či je hra vyhratá alebo prehratá. Možné stavy, ktoré možu nastať po dokončení kola:

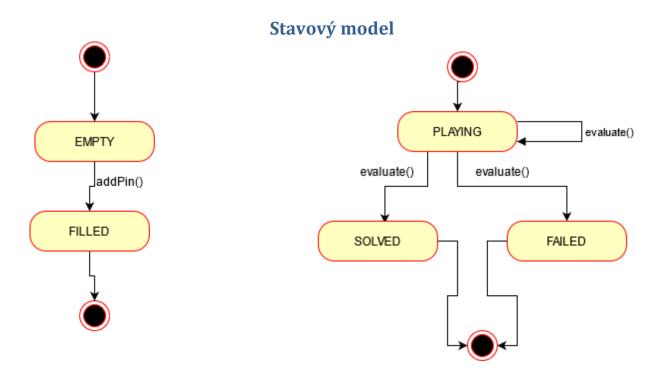
- 1. Hráč uhádol počítačom vygenerovanú kombináciu. Koniec hry výhra
- 2. Hráč neuhádol počítačom vygenerovanú kombinúciu a uplynul počet určených kôl, ktoré na to mal. **Koniec hry prehra**
- 3. Hráč neuhádol počítačom vygenerovanú kombinúciu no ešte nevyplítval všetky pokusy. **Hra pokračuje**

#### Ťah hráča

Ťah hráča spočíva vo výbere jednej z nasledujúcich akcií:

- 1. Priradenie pinu (addPin()), do jednej z dierok
- 2. Potvrdenie svojho výberu, keď uložil všetky 4 piny (evaluate())
- 3. Požiadanie hry o výpis historie jeho tipov (printHistory())

#### Návrhový model <<enumeration>> <u>PinColor</u> +RED +BLUE +GREEN +YELLOW +PINK Pin +BROWN +BLACK color: PinColor Mastermind +WHITE main(args String) +EMPTY getColor(): Pin.Color Game ConsoleUl playerHoles: Hole[4] Hole evaluationHoles: Hole[4] start() stant() printHistory(int, int) printHistory(int) communicateWithPlayer() history: Hole[9][4] pin: Pin 8 1 generate() getPin(): Pin setPin(Pin) evaluate(): Hole[4] putPin(PinColor, int) <<enumeration>> Game.State <<enumeration>> +PLAYING Hole.State +FAILED +EMPTY +SOLVED



+FILLED