《斗兽棋》开发说明文档

1. **介绍**

1.1 目的：本文档的目的是提供斗兽棋开发说明，以帮助团队理解并实现斗兽棋游戏。文档将介绍游戏规则、技术要求、功能设计、测试与调试等方面的内容，旨在为开发人员提供一个全面的指南，帮助他们有效地开发和构建斗兽棋游戏。通过本文档，读者将能够了解游戏规则、掌握游戏开发所需的技术要求、了解游戏功能设计的方法和实现细节，并获得测试和调试游戏的技巧和建议。最终目的是使开发人员能够成功地开发出一个功能完善、稳定可靠的斗兽棋游戏。

1.2 背景：

斗兽棋是一种古老而受欢迎的棋类游戏，起源于中国。它被认为是中国最早的棋类游戏之一，具有悠久的历史和深厚的文化底蕴。

斗兽棋的起源可以追溯到公元前2世纪的中国西汉朝。传说中，该游戏最初是由汉朝皇帝刘备创造的，他为了培养军事战略能力而设计了这个游戏。斗兽棋的规则和棋盘都是根据古代战争中的策略和阵型而设计的。

随着时间的推移，斗兽棋逐渐传播到世界各地，成为一种受欢迎的棋类游戏。如今，在计算机和移动设备的普及下，斗兽棋也逐渐以电子游戏的形式出现，为玩家提供了更加便捷和多样化的游戏体验。

1.3 目标用户：玩家

2.**游戏规则**

2.1 游戏目标：

- 胜利条件：进入对方兽穴，或将对方的棋子吃光。

- 玩法规则：玩家通过移动自己的棋子来攻击和防守对方的棋子，通过战斗和策略来达成胜利条件。

2.2 棋子及其能力：

- 鼠：可以在陆地上移动，也可以进入河流。战胜象。

- 猫：可以在陆地上移动，不能跳过其他棋子。战胜老鼠。

- 狗：可以在陆地上移动，不能跳过其他棋子。战胜猫，老鼠。

- 狼：可以在陆地上移动，不能跳过其他棋子。战胜狗，猫，老鼠。

- 豹：可以在陆地上移动，不能跳过其他棋子。战胜狼，狗，猫，老鼠。

- 虎：可以在陆地移动和河上跳跃，不能跳过其他棋子。战胜豹，狼，狗，猫，老鼠。

- 狮：可以在陆地移动和河上跳跃，不能跳过其他棋子。战胜虎，豹，狼，狗，猫，老鼠。

- 象：可以在陆地上移动，不能跳过其他棋子。战胜狮，虎，豹，狼，狗，猫。

2.3 游戏流程：

1. 摆放棋子：红方和蓝方轮流摆放自己的棋子，每方依次放置一只棋子，共摆放16只棋子。

2. 移动棋子：轮流移动自己的棋子，每次只能移动一只棋子到相邻的空格中（前、后、左、右）。

3. 战斗：如果移动的棋子进入敌方占据的格子，战斗开始。两只棋子按照它们的实力进行比较，实力较弱的棋子被吃掉。

4. 胜利判定：游戏继续进行，直到达成胜利条件之一（进入敌方兽穴或将敌方棋子吃光）。

通过以上规则，玩家可以进行一局斗兽棋游戏。交替移动棋子、战斗和利用每个棋子的特殊能力将是游戏中取胜的关键。玩家需要思考每一步的移动和战术选择，以最终击败对方并达到游戏目标。

3. **技术要求**

3.1 开发环境：

编程语言：开发斗兽棋所需的编程语言选择包括C++,QML。

3.2 前端开发：根据需求和个人技术能力，利用素材图片结合QML放置到游戏界面上去

3.3 后端开发：分别将所有棋子统一用一个Animals组件，棋子的事件处理用一个Action组件，棋盘的绘制用一个Chessboard组件，棋盘中的所有点击事件用一个Chessmove组件

4. 功能设计

4.1 界面及交互方式：

* 游戏界面：设计一个简介、美观的游戏界面，包括棋盘、棋子、按钮和其他相关元素。
* 交互方式：提供用户友好的交互方式，如点击棋子选择、移动棋子、展示战斗结果等操作，以及按钮控制游戏的开始、继续、退出等功能。

4.2 游戏逻辑：分为五个组件

**Animals组件**:每个棋子都有property：lastX、lastY（用于记录棋子当前的位置）， selected（用于记录棋子是否被选中），actar（用于区分红蓝方），type用于棋子的类型（1——8分别表示，老鼠，猫，狗，狼，豹子，老虎，狮子，大象）。

**Chessmove组件:**棋盘有

properties： selected（用于记录棋盘是否被选中）sequence（用于判断双方下棋的顺序，每次一方只能移动己方的一枚棋子）。

**Functions**:

function initAnimalsData（）：用于初始化每个动物棋子的property；

function initChessboardData（）：用于初始化棋盘的property;

function init（）调用前两个function实现整局游戏初始化;

function whitchactar（）：用于判断游戏回合，红方只能在红方回合移动棋子，蓝方反之;

function isvectory（）：用于判断棋子是否进入对方巢穴;

function vectory（）：调用isvectory（）判断棋子进入巢穴或一方没有棋子了;

function isinRiver（）：判断狮虎是否能跳河;

function isRiver（）：判断老鼠是否在河里；

function eating（）：提取eat（）中重复的代码；

function eat（）：实现棋子的互吃

function caculating（）：提取caculate（）中重复的代码;

function caculate（）：实现棋子的移动；

**Action组件：**里面封装了每个棋子都有的点击行为，只要点击了棋子就更新该棋子的当前位置

**Bground组件**：绘制背景以及两个小人

**StartGame组件：**开始游戏界面,开始游戏按钮，退出游戏按钮，以及查看游戏规则按钮

* **棋子移动**：

**原理**：首先玩家点击己方的棋子使棋子被选中（Animals.selected=true），同时记录棋子的当前位置 (Animals.lastX=(Math.floor(Animals.x/100)\*100)+50,

Animals.lastY=(Math.floor(Animals.y/100)\*100)+50，）；然后点击棋盘上玩家想要移动到的位置。如果两次的点击都符合斗兽棋的规则，则首先计算出被点击棋子的位置与棋盘点击位置的距离(dx =(Math.floor(handler.point.position.x/100)\*100)+50-Animals.lastX,dy=(Math.floor(handler.point.position.y/100)\*100)+50-Animals.lastY)，然后棋子的当前位置加上距离为棋子的新位置(Animals.x+=dx, Animals.y+=dy)，同时更新棋子和棋盘的状态包括棋子当前位置(Animals.lastX=Animals.x+50,Animals.lastY=Animals.y+50），还有棋子被选中（Animals.selected=false）,棋盘的顺序加一(Chessboard.sequence+=1),棋盘的被选中（Chessboard.selected=false）

* **棋子战斗**：

**原理：**首先玩家点击己方棋子使棋子被选中（Animals.selected=true），同时记录棋子的当前位置 (Animals.lastX=(Math.floor(Animals.x/100)\*100)+50,

Animals.lastY=(Math.floor(Animals.y/100)\*100)+50，）。然后点击棋盘上想要吃掉的敌方棋子。如果两次的点击都符合斗兽棋的规则，则首先计算出被点击己方棋子(parent)的位置与被点击敌方棋子(Animals)位置的距离(cx=(Math.floor(parent.x/100)

\*100)+50-Animals.lastX,cy=(Math.floor(parent.y/100)\*100)+50-Animals.lastY)，然后己方棋子的位置加上距离为己方棋子的新位置（Animals.x+=cx, Animals.y+=cy）,同时敌方棋子被吃（parent.visible=false）,同时更新棋子的状态两颗棋子的(selected=false),己方棋子的当前位置更新(Animals.lastX=Animals.x+50,

Animals.lastY=Animals.y+50）,棋盘的顺序加一(chessboard+=1)，棋盘的被选中取消(chessboard.selected=false)

* 胜负判定：根据胜利条件判断游戏的胜负结果，如任意一颗棋子吃掉对方的最后一颗棋子、进入对方巢穴等。

4.3 游戏状态：说明游戏状态的管理和转换，包括开始新游戏、退出游戏等

5. 结尾

总结：本次斗兽棋小游戏开发的难点在于实现其中的特有的游戏规则：比如最开始困扰我们的是棋子的移动，后来经过我们仔细琢磨其中的下棋步骤，先点击己方棋子，再点击将要移动的位置，分别获取两处的位置然后将棋子的位置改变；还有分为游戏双方，一次只能一方移动棋子，然后轮到下一方，这样我们就引入了sequence标记顺序；还有实现棋子的吃，一般情况下大吃小，特定情况下小才能吃大；还有特定的老鼠才能下河；

特定的虎，狮才能跳过河，但是如果路径上有老鼠的话就不能不能跳……等等一些列问题。最后经过我们小组的讨论和课后查阅资料实现了这样一些规则